Oracle Machine Learning for R 2.0

October 30, 2023

R topics documented:

`OREbase-package` *Oracle R Enterprise Base Functions*

Description

The **OREbase** package implements the Oracle R Enterprise transparency layer for R's **base** package.

Details

Package:	OREbase
Type:	Package
Version:	2.0
Date:	2023-10-16
Depends:	methods, utils, OREcommon
Suggests:	DBI, ROracle (>= 1.1-12)
License:	file LICENSE
Collate:	classes.R generics.R cli-hive.R query.R internal.R query-oracle.R query-hive.R query-impala.R mapping.R r
LazyLoad:	yes

URL: http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html
https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.raw, ore.vector

Examples

```
# Transparency layer function overloads of the base package
intersect(ls("package:OREbase"), ls("package:base"))

# Transparency layer function overloads of the methods package
intersect(ls("package:OREbase"), ls("package:methods"))
```

`OREdm-package` *Oracle R Enterprise Data Mining Functions*

Description

The **OREdm** package provides access to the in-database data mining functionality of Oracle Database. The Oracle R Enterprise Data Mining package contains several data mining and data analysis algorithms for classification, regression, clustering, attribute importance, and anomaly detection.

Details

Package:	OREdm
Type:	Package
Version:	2.0
Date:	2023-10-16
Depends:	OREbase (>= 2.0)
Imports:	utils, methods, lattice, stats, ROracle, OREstats
Suggests:	arules
License:	file LICENSE
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

rules	Predicate Rule Data for Clusters or Association Rules Models

The functions above are used to build data mining models in Oracle Database. Each function implicitly uses the Oracle R Enterprise database connection and takes a formula object to specify the predictors (terms) and target (response).

To use this package with Oracle Database 11g, Oracle Data Mining must be enabled. To use this package with Oracle Database 12c, the Oracle Advanced Analytics option must be enabled.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts 11g Release 2 (11.2) http://download.oracle.com/docs/cd/E11882_01/datamine.112/e16808/toc.htm

See Also

ore.connect

OREdplyr-package *Oracle R Enterprise Data Manipulation Functions*

Description

The **OREdplyr** package contains functions for data manipulation.

Details

Package:	OREdplyr
Type:	Package
Version:	2.0
Date:	2023-10-25
Depends:	OREbase
Suggests:	dplyr, magrittr
Imports:	methods
License:	file LICENSE
Collate:	generics.R internal.R methods-default.R methods-ore.vector.R methods-ore.frame.R zzz.R
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

desc	Descending Order
distinct	Select Distinct Rows
group_by	Group Rows by One or More Columns
join	Join Two ore.frame Objects
n	The number of rows in the current group
n_distinct	Number of Unique Values
nth	Get the nth Value
ranking	Various Ranking Functions
sample	Sample Rows
select	Select, Rename, Arrange, Filter, Mutate, or Transmute
slice	Select Rows by Positions
summarise	Summarise Columns by Aggregate Functions
tally	Counts or Tallies Rows by Group
top_n	Select Top n Rows

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

desc, distinct, group_by, join, n, n_distinct, nth, ranking, sample, select, slice, summarise, tally, top_n

---

OREds-package *Oracle R Enterprise Datastore Functions*

Description

The **OREds** package contains functions for managing R objects stored within the Oracle database.

Details

Package:	OREds
Type:	Package
Version:	2.0
Date:	2023-10-16
Depends:	OREbase
Suggests:	rpart
Imports:	methods, OREcommon, ROracle (>= 1.1-12), utils
License:	file LICENSE
Collate:	persist.R
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.delete, ore.datastore, ore.datastoreSummary

---

OREeda-package *Oracle R Enterprise Exploratory Data Analysis Functions*

Description

The **OREeda** package contains functions that allow different types of analysis of numeric columns in the input object of type ore.frame with flexible aggregation, ranking, cross-tabulation, and sorting functionality.

Details

Package:	OREeda
Type:	Package
Version:	2.0
Date:	2023-10-17
Depends:	OREbase (>= 2.0)
Imports:	methods, stats, DBI, OREbase, OREembed, OREstats
Suggests:	ROracle (>= 1.1-12), OREgraphics
Collate:	oreUtils.R orePrvt.R oreParams.R oreDLL.R oreCorr.R oreCrosstab.R oreCrossval.R oreExtend.R oreConve
License:	file LICENSE
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

```
ore.crosstab            Oracle R Enterprise Crosstabulations
ore.esm                 Oracle R Enterprise Time Series Exponential Smoothing Mo
ore.freq                Oracle R Enterprise Frequency Analysis
ore.rank                Oracle R Enterprise Data Ranking
ore.sort                Oracle R Enterprise Data Sorting
ore.summary             Oracle R Enterprise Data Summary
ore.univariate          Oracle R Enterprise Univariate Summaries
```

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.corr, ore.crosstab, ore.freq, ore.rank, ore.sort, ore.summary, ore.univariate

---

`OREembed-package` *Oracle R Enterprise Embedded R Functions*

Description

The **OREembed** package contains functions for using the Oracle R Enterprise embedded R functionality.

Details

Package:	OREembed
Type:	Package
Version:	2.0
Date:	2023-10-17
Depends:	OREbase
Suggests:	rpart
Imports:	methods, OREcommon, OREds, ROracle (>= 1.1-12), grDevices, grid, png, utils
License:	file LICENSE
Collate:	classes.R generics.R internal.R methods-ore.R methods-ore.object.R methods-ore.list.R methods-ore-data-in
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

```
OREembed-package        Oracle R Enterprise Embedded R Functions
ore.doEval              Oracle R Enterprise Embedded R Script Execution
                        Functions
ore.list-class          Class ore.list
ore.object-class        Class ore.object
```

ore-data-image-class	Class ore-data-image
ore.scriptCreate	Oracle R Enterprise Embedded R Script Creation or Destruction Functions

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.doEval, ore.list, ore.object, ore-data-image

OREgraphics-package

Oracle R Enterprise Graphics Functions

Description

The **OREgraphics** package implements the Oracle R Enterprise transparency layer for R's **graphics** package.

Details

Package:	OREgraphics
Type:	Package
Version:	2.0
Date:	2023-10-17
Depends:	methods, grDevices, graphics, OREbase (>= 2.0)
Imports:	stats, OREstats
License:	file LICENSE
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

ore-plot	'plot' methods for Oracle R Enterprise
ore-points	'points' methods for Oracle R Enterprise
ore-polygon	'polygon' methods for Oracle R Enterprise
ore-polypath	'polypath' methods for Oracle R Enterprise
ore-rug	'rug' methods for Oracle R Enterprise
ore-segments	'segments' methods for Oracle R Enterprise
ore-smoothScatter	'smoothScatter' methods for Oracle R Enterprise
ore-sunflowerplot	'sunflowerplot' methods for Oracle R Enterprise
ore-symbols	'symbols' methods for Oracle R Enterprise
ore-text	'text' methods for Oracle R Enterprise
ore-xspline	'xspline' methods for Oracle R Enterprise
ore-xy.coords	'xy.coords' methods for Oracle R Enterprise

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

OREpredict-package *Oracle R Enterprise Model Prediction Functions*

Description

The **OREpredict** package contains functions for generating model predictions using standard R models.

Details

Package:	OREpredict
Type:	Package
Version:	2.0
Date:	2023-10-17
Depends:	utils, methods, stats, OREbase (>= 2.0), OREmodels
Imports:	OREstats
Suggests:	MASS, rpart, nnet
License:	file LICENSE
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

```
ore.predict-prcomp        Method for prcomp models
ore.predict-princomp      Method for princomp models
ore.predict-rpart         Method for rpart models
ore.predict-nnet          Method for nnet models
ore.predict-multinom      Method for multinom models
ore.predict-ore.lm        Method for ore.lm models
```

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

`OREstats-package` *Oracle R Enterprise Statistical Functions*

Description

The **OREstats** package implements the Oracle R Enterprise transparency layer for R's **stats** and **MASS** packages.

Details

Package:	OREstats
Type:	Package
Version:	2.0
Date:	2023-10-17
Depends:	methods, stats, MASS, OREbase (>= 2.0)
Imports:	OREembed
License:	file LICENSE
LazyLoad:	yes
URL:	http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

```
ore.factor-reorder          Reorder Levels of an \code{ore.factor} Object
ore.frame-factanal          Oracle R Enterprise Factor Analysis
ore.frame-model.frame       Oracle R Enterprise Modeling Framework
ore.frame-princomp          Oracle R Enterprise Principal Components Analysis
ore.frame-prcomp            Oracle R Enterprise Principal Components Analysis
ore.frame-svd               Oracle R Enterprise Singular Value Decomposition
ore.getXlevels              Factor Levels for an Oracle R Enterprise
                            Model Matrix
ore.number-Distributions    Oracle R Enterprise Distribution Functions
ore.number-fitdistr         Oracle R Enterprise Maximum Likelihood
                            Estimation of Univariate Distributions
ore.number-multivar         Oracle R Enterprise Multivariate Statistics
```

ore.number-univar	Oracle R Enterprise Univariate Statistics
ore.roll	Oracle R Enterprise Rolling Window Aggregates
ore.tblmatrix-svd	Oracle R Enterprise Singular Value Decomposition
ore.vecmatrix-svd	Oracle R Enterprise Singular Value Decomposition
ore.vector-aggregate	Oracle R Enterprise Univariate Aggregation by a Set of Grouping Variables
ore.vector-ave	Oracle R Enterprise Univariate Operations within a Set of Grouping Variables
ore.vector-binom.test	Oracle R Enterprise Exact Binomial Test
ore.vector-chisq.test	Oracle R Enterprise Pearson's Chi-squared Test for Count Data
ore.vector-ks.test	Oracle R Enterprise Kolmogorov-Smirnov Tests
ore.vector-prop.test	Oracle R Enterprise Test of Equal or Given Proportions
ore.vector-t.test	Oracle R Enterprise Student's t-Test
ore.vector-var.test	Oracle R Enterprise F Test to Compare Two Variances
ore.vector-wilcox.test	Oracle R Enterprise Wilcoxon Rank Sum and Signed Rank Tests

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

```
# Transparency layer function overloads of the stats package
intersect(ls("package:OREstats"), ls("package:stats"))
```

ORExml-package *Oracle R Enterprise XML String Generation Functions*

Description

The **ORExml** package contains functions for converting R objects to XML.

Details

Package:	ORExml
Type:	Package
Version:	2.0
Date:	2023-10-25
Depends:	OREbase
Imports:	methods
License:	file LICENSE
Collate:	ORExml.R
LazyLoad:	yes

URL: http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/documentation/index.html
https://community.oracle.com/community/developer/english/business_intelligence/data_warehousing/r

Index:

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.toXML

---

arrows-methods *Add Arrows to a Plot*

Description

Draws arrows between pairs of points.

Usage

```
  ## S4 method for signature 'ore.vector'
arrows(x0, y0, x1 = x0, y1 = y0, length = 0.25,
         angle = 30, code = 2, col = par("fg"), lty = par("lty"),
         lwd = par("lwd"), ...)
```

Arguments

x0,y0,x1,y1	ore.number objects.
length,angle,code,col,lty,lwd,...	See description in arrows.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

arrows

as.ore *Oracle R Enterprise Coercion Functions*

Description

Coerces objects to ore class types.

Usage

```
as.ore(x, ...)
as.ore.vector(x, mode = "any")
as.ore.logical(x, ...)
as.ore.integer(x, ...)
as.ore.numeric(x, ...)
as.ore.character(x, ...)
as.ore.factor(x, ...)
as.ore.date(x, ...)
as.ore.datetime(x, ...)
as.ore.difftime(x, ...)
as.ore.frame(x, ...)
as.ore.matrix(x, ...)
```

Arguments

x	An object.
mode	A character string specifying an atomic mode or "any". The supported atomic modes are "logical", "integer", "numeric" (synonym "double"), and "character".
...	Additional arguments.

Details

The as.ore function coerces in-memory R objects to ore objects. It is similar to the ore.push function, but whereas ore.push may not create an ore object if no mapping exists, the as.ore function will throw an error if it cannot create an ore object. See ore.push for more information on the mapping from in-memory R objects to ore objects.

Value

An ore object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.push, is.ore, as.vector

Examples

```
df <- data.frame(A = 1:26, B = letters)
dim(df)
class(df)
oreDF <- as.ore(df)
class(oreDF)
dim(oreDF)
head(oreDF)

vec <- 1:20
class(vec)
oreVec <- as.ore(vec)
class(oreVec)
```

---

binom.test-methods *Oracle R Enterprise Exact Binomial Test*

Description

Performs an exact test of a simple null hypothesis about the probability of success in a Bernoulli experiment.

Usage

```
  ## S4 method for signature 'ore.vector'
binom.test(x, n, p = 0.5,
           alternative = c("two.sided", "less", "greater"),
           conf.level = 0.95)
```

Arguments

x	An ore.vector object with exactly two distinct values.
n	Argument not supported.
p	A numeric value specifying the hypothesized probability of success.
alternative	A character string specifying the alternative hypothesis and must be one of "two.sided", "greater" or "less".
conf.level	A numeric value specifying the confidence level for the returned confidence interval.

Details

The binom.test method in **OREstats** differs from the binom.test function in the **stats** package in that the ore.vector object represents the "raw" successes and failures instead of the aggregate counts of those successes and failures.

Value

Returns an binom.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Clopper, C. J. & Pearson, E. S. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, **26**, 404–413.

William J. Conover (1971), *Practical nonparametric statistics*. New York: John Wiley & Sons. Pages 97–104.

Myles Hollander & Douglas A. Wolfe (1973), *Nonparametric Statistical Methods.* New York: John Wiley & Sons. Pages 15–22.

Oracle R Enterprise

See Also

binom.test

---

boxplot.ore *Create Box Plots*

Description

Creates Box-and-Whisker Plots

Usage

```
  ## S3 method for class 'ore'
boxplot(x, ..., range = 1.5, width = NULL, varwidth = FALSE,
          notch = FALSE, outline = TRUE, names, plot = TRUE,
          border = par("fg"), col = NULL, log = "",
          pars = list(boxwex = 0.8, staplewex = 0.5, outwex = 0.5),
          horizontal = FALSE, add = FALSE, at = NULL)
```

Arguments

x	An ore.number object, a list of ore.number objects or a formula, e.g. y ~ grp, where y is an ore.number object and grp is an ore.factor object.
...,range,width,varwidth,notch,outline,names,plot,border,col,log,pars,horizontal	See description in boxplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

boxplot

cdplot.ore *Conditional Density Plots*

Description

Computes and plots conditional densities.

Usage

```
  ## S3 method for class 'ore'
cdplot(x, y, plot = TRUE, tol.ylab = 0.05, ylevels = NULL,
         bw = "nrd0", n = 512, from = NULL, to = NULL, col = NULL,
         border = 1, main = "", xlab = NULL, ylab = NULL,
         yaxlabels = NULL, xlim = NULL, ylim = c(0, 1), ...)
```

Arguments

x	An ore.number object.
y	An ore.factor object.
plot,tol.ylab,ylevels,bw,n,from,to,col,border,main,xlab,ylab,yaxlabels,xlim,ylim	See description in cdplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

cdplot

chisq.test-methods *Oracle R Enterprise Pearson's Chi-squared Test for Count Data*

Description

Performs chi-squared contingency table tests and goodness-of-fit tests.

Usage

```
  ## S4 method for signature 'ore.vector'
chisq.test(x, y = NULL, correct = TRUE,
           p = rep(1/length(x), length(x)), rescale.p = FALSE,
           simulate.p.value = FALSE, B = 2000)
```

Arguments

x	An ore.vector object.
y	An ore.vector object or a NULL value.
correct	A logical value indicating whether to apply continuity correction when computing the test statistic for 2 by 2 tables: one half is subtracted from all $\|O - E\|$ differences. No correction is done if simulate.p.value = TRUE.
p	If argument y = NULL, a vector of probabilities of the same length as argument x.
rescale.p	A logical value indicating if p should be rescaled (if necessary) to sum to 1.
simulate.p.value	A logical value indicating whether to compute p-values by Monte Carlo simulation.
B	An integer value specifying the number of replicates to use in the Monte Carlo test.

Value

Returns an chisq.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Hope, A. C. A. (1968) A simplified Monte Carlo significance test procedure. *J. Roy, Statist. Soc. B* **30**, 582–598.

Patefield, W. M. (1981) Algorithm AS159. An efficient method of generating r x c tables with given row and column totals. *Applied Statistics* **30**, 91–97.

Agresti, A. (2007) *An Introduction to Categorical Data Analysis, 2nd ed.*, New York: John Wiley & Sons. Page 38.

Oracle R Enterprise

See Also

chisq.test

---

clusterhists *Cluster Histogram Data from Clustering Models*

Description

The generic function clusterhists returns cluster histogram data, which may be used to generate histograms.

Usage

```
clusterhists(object,...)

## S3 method for class 'ore.odmKMeans'
clusterhists(object,...)
## S3 method for class 'ore.odmOC'
clusterhists(object,...)
## S3 method for class 'ore.odmEM'
clusterhists(object,...)
```

Arguments

object	Object for which cluster histogram data is desired.
...	Additional arguments affecting the result.

Details

The function clusterhists provides the histogram data for each variable for each cluster.

This function generates a data.frame with histogram data for each cluster and variable combination in the model. Numerical variables are binned. A label is produced combining the range of the bin for numerical variables. The label for categorical variables contains the value.

Value

The function clusterhists returns a data.frame with columns:

cluster.id	The numeric identifier associated with a cluster.
variable	The character name of the variable.
bin.id	The numeric identifier of a bin for a variable in the range of 1 to N where N is the number of bins generated if the variable is numeric, or the number of categories if the variable is categorical.
lower.bound	For a numeric variable, the lower bound of the bin.
upper.bound	For a numeric variable, the upper bound of the bin.
label	For a numeric variable, the concatenation of lower bound and upper bound. For a categorical variable, the category string.
count	The number of rows with this value range in the cluster for this variable, or the probability of the values in this value range appearing in the cluster for this variable.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmKMeans, ore.odmOC ore.odmEM

Examples

```
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
         matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
X <- ore.push (data.frame(x))

km.mod <- NULL
km.mod <- ore.odmKMeans(~., X, num.centers=2)
clusterhists(km.mod)
```

---

desc *Descending Order*

Description

Sorts an ore.number, ore.factor, or ore.character object in descending order.

Usage

```
desc(x)
```

Arguments

x	The ore.vector object to transform.

Examples

```
desc(ore.push(1:10))
desc(ore.push(factor(letters)))
```

---

distinct *Select Distinct Rows*

Description

Retains unique rows from an input ore.frame object over the specified columns.

Usage

```
distinct(.data, ...)

distinct_(.data, ..., .dots)
```

Arguments

.data	An ore.frame object.
...	Columns to determine the uniqueness.
.dots	Used to work around non-standard evaluation. See distinct_ for details.

Examples

```
df <- data.frame(
  x = sample(10, 100, rep = TRUE),
  y = sample(10, 100, rep = TRUE)
)
DF <- ore.push(df)
nrow(DF)
nrow(distinct(DF))
arrange(distinct(DF, x), x)
arrange(distinct(DF, y), y)

# Use distinct on computed variables
arrange(distinct(DF, diff = abs(x - y)), diff)
```

---

fitdistr-methods *Oracle R Enterprise Maximum Likelihood Estimation of Univariate Distributions*

Description

Maximum-likelihood fitting of univariate distributions, allowing parameters to be held fixed if desired.

Usage

```
  ## S4 method for signature 'ore.number'
fitdistr(x, densfun, start, ...)
```

Arguments

x	An ore.number object.
densfun	A character string specifying the density function; one of "beta", "cauchy", "chi-squared", "exponential", "f", "gamma", "geometric", "log-normal", "lognormal", "logistic", "negative binomial", "normal", "Poisson", "t" and "weibull".
start	A named list giving the parameters to be optimized with initial values. This can be omitted for some of the named distributions and must be for others.
...	Additional parameters, either for 'densfun' or for 'optim'.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

fitdistr

---

group_by *Group Rows by One or More Columns*

Description

group_by Groups an ore.frame object over the specified columns. Functions summarise, top_n, count can perform on such an ore.frame object group-wise. ungroup Drops the grouping from the input ore.frame object. group_size Lists the number of rows in each group. groups Shows the names of the grouping columns. n_groups Gives the number of groups.

Usage

```
group_by(.data, ..., add = FALSE)
group_by_(.data, ..., .dots, add = FALSE)

groups(x)
ungroup(x)

group_size(x)

n_groups(x)
```

Arguments

.data	An ore.frame object.
x	A grouped ore.frame object.
...	Columns to group by.
add	By default, add = FALSE, group_by overrides the existing groups. Use add = TRUE to add new groups.
.dots	Used to work around non-standard evaluation. See group_by_ for details.

See Also

summarise

Examples

```
MTCARS <- ore.push(mtcars)
by_cyl <- group_by(MTCARS, cyl)
arrange(summarise(by_cyl, mean(disp), mean(hp)), cyl)

# summarise drops one layer of grouping
by_vs_am <- group_by(MTCARS, vs, am)
by_vs <- summarise(by_vs_am, n = n())
arrange(by_vs, vs, am)
arrange(summarise(by_vs, n = sum(n)), vs)

# remove grouping
summarise(ungroup(by_vs), n = sum(n))

# group by expressions with mutate
arrange(group_size(group_by(mutate(MTCARS, vsam = vs + am), vsam)), vsam)

# rename the grouping column
groups(rename(group_by(MTCARS, vs), vs2 = vs))

# add more grouping columns
groups(group_by(by_cyl, vs, am))
groups(group_by(by_cyl, vs, am, add = TRUE))

# Duplicate groups are dropped
groups(group_by(by_cyl, cyl, cyl))

library(magrittr)
by_cyl_gear_carb <- MTCARS %>% group_by(cyl, gear, carb)
n_groups(by_cyl_gear_carb)
arrange(group_size(by_cyl_gear_carb), cyl, gear, carb)

by_cyl <- MTCARS %>% group_by(cyl)

# number of groups
n_groups(by_cyl)

# size of each group
arrange(group_size(by_cyl), cyl)
```

---

hist.ore *Histograms*

Description

Computes histograms.

Usage

```
  ## S3 method for class 'ore'
hist(x, breaks = "Sturges", freq = NULL,
       probability = !freq, density = NULL,
       angle = 45, col = NULL, border = NULL,
       main = paste("Histogram of" , xname),
```

```
          xlim = range(breaks), ylim = NULL,
          xlab = xname, ylab, axes = TRUE,
          plot = TRUE, labels = FALSE,
          warn.unused = TRUE, ...)
```

Arguments

x	An ore.number object.
breaks	Either a single number giving the number of cells for the histogram or one of the following character strings: "Sturges", "Scott", "FD", or "Freedman-Diaconis". See Details in hist.
freq,probability,density,angle,col,border,main,xlim,ylim,xlab,ylab,axes,plot,lab	See description in hist.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

hist

---

identify.ore *Identify Points in a Scatter Plot*

Description

Interactively marks positions of a graphic.

Usage

```
  ## S3 method for class 'ore'
identify(x, y = NULL, labels = seq_along(x), pos = FALSE,
           n = length(x), plot = TRUE, atpen = FALSE,
           offset = 0.5, tolerance = 0.25, ...)
```

Arguments

x,y	ore.number objects.
labels,pos,n,plot,atpen,offset,tolerance,...	See description in identify.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

identify

`is.ore` *Oracle R Enterprise Type Checking Functions*

Description

Returns logical value `TRUE` if the `x` argument is an object of the type named in the function; otherwise returns logical value `FALSE`.

Usage

```
is.ore(x)
is.ore.vector(x)
is.ore.logical(x)
is.ore.integer(x)
is.ore.numeric(x)
is.ore.character(x)
is.ore.factor(x)
is.ore.date(x)
is.ore.datetime(x)
is.ore.difftime(x)
is.ore.frame(x)
is.ore.matrix(x)
```

Arguments

`x`	An object.

Value

The logical value `TRUE` if the `x` argument is an `ore` object of the specified type, or the logical value `FALSE` otherwise.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore`, `as.ore`, `is.vector`

Examples

```
df <- data.frame(A = 1:26, B = letters)
is.ore(df)

oreDF <- as.ore(df)
is.ore(oreDF)
```

---

join *Join Two* ore.frame *Objects by the Specified Columns*

Description

inner_join Returns all combination of rows from x and y over matched columns. left_join Returns rows from inner_join plus rows from x that do not match with y. For unmatched rows of x, NA is returned. right_join Returns rows from inner_join plus rows from y that do not match with x. For unmatched rows of y, NA is returned. full_join Returns the union of rows from left_join and right_join.

Usage

```
inner_join(x, y, by = NULL)

left_join(x, y, by = NULL)

right_join(x, y, by = NULL)

full_join(x, y, by = NULL)
```

Arguments

x,y	The ore.frame objects to join.
by	A character vector of column names to join by. The default is NULL. The join function does a natural join, using all columns with common names across the two tables. To join by different columns on x and y, use a named vector string. For example, by = c("a" = "b") will match column a in x with column b in y.

Examples

```
MTCARS <- ore.push(mtcars)
M1 <- filter(select(MTCARS, mpg, cyl, carb), carb < 6L)
M2 <- filter(select(MTCARS, cyl, hp, carb), carb > 2L)

names(inner_join(M1, M2))
nrow(inner_join(M1, M2))
nrow(left_join(M1, M2))
nrow(right_join(M1, M2))
nrow(full_join(M1, M2))

names(M2) <- c("cyl", "hp", "carb2")
names(inner_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(inner_join(M1, M2, by = c("cyl", carb="carb2")))
```

```
nrow(left_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(right_join(M1, M2, by = c("cyl", carb="carb2")))
nrow(full_join(M1, M2, by = c("cyl", carb="carb2")))
```

`ks.test-methods` *Oracle R Enterprise Kolmogorov-Smirnov Tests*

Description

Performs a one- or two-sample Kolmogorov-Smirnov test.

Usage

```
  # one sample test
  ## S4 method for signature 'ore.number,character'
ks.test(x, y, ...,
        alternative = c("two.sided", "less", "greater"),
        exact = NULL)

  # two sample test
  ## S4 method for signature 'ore.number,ore.number'
ks.test(x, y, ...,
        alternative = c("two.sided", "less", "greater"),
        exact = NULL)

  # two sample test; use x as a grouping variable
  ## S4 method for signature 'ore.vector,ore.number'
ks.test(x, y, ...,
        alternative = c("two.sided", "less", "greater"),
        exact = NULL)
```

Arguments

`x`	An `ore.number` object or an `ore.factor` object with exactly two levels.
`y`	Either an `ore.number` object or one of the following character strings specifying a cumulative distribution function: `"pexp"`, `"pnorm"`, `"ppois"`, `"punif"`, `"pweibull"`.
`...`	Parameters of the distribution specified (as a character string) by argument `y`.
`alternative`	A character string specifying the alternative hypothesis and must be one of `"two.sided"` (default), `"less"`, or `"greater"`.
`exact`	Argument not supported.

Value

Returns an `ks.test` `htest` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Z. W. Birnbaum and Fred H. Tingey (1951), One-sided confidence contours for probability distribution functions. *The Annals of Mathematical Statistics*, **22**/4, 592–596.

William J. Conover (1971), *Practical Nonparametric Statistics*. New York: John Wiley & Sons. Pages 295–301 (one-sample Kolmogorov test), 309–314 (two-sample Smirnov test).

Durbin, J. (1973) *Distribution theory for tests based on the sample distribution function*. SIAM.

George Marsaglia, Wai Wan Tsang and Jingbo Wang (2003), Evaluating Kolmogorov's distribution. *Journal of Statistical Software*, **8**/18. http://www.jstatsoft.org/v08/i18/.

Oracle R Enterprise

See Also

ks.test

---

lines.ore *Add Connected Line Segments to a Plot*

Description

Draws line segments.

Usage

```
  ## S3 method for class 'ore'
lines(x, y = NULL, type = "l", ...)
```

Arguments

x,y	ore.number objects.
type,...	See description in lines.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

lines

n — *The number of rows in the current group*

Description

This function can only be used from within summarise, mutate and filter to get the number of rows in the group.

Usage

```
n()
```

Examples

```
MTCARS <- ore.push(mtcars)
mtcars_cyl <- group_by(MTCARS, cyl)
arrange(summarise(mtcars_cyl, n = n()), cyl)
mutate(mtcars_cyl, n = n())
filter(mtcars_cyl, n() <= 10L)
```

nth — *Get the First, Last or nth Value from an Ordered* ore.vector *Object*

Description

Obtains the value at the specified position in the order.

Usage

```
nth(x, n, order_by = NULL, default = NULL)

first(x, order_by = NULL, default = NULL)

last(x, order_by = NULL, default = NULL)
```

Arguments

x	An ordered ore.vector object.
n	An integer specifying the position of a value in the order.
order_by	An ore.vector or column used to determine the order.
default	not used.

Value

A scalar ore.number, ore.character, or ore.factor value.

Examples

```
X <- ore.push(1:10)
Y <- ore.push(10:1)

nth(X, 2)
nth(X, 2, Y)

last(X)
last(X, order_by = Y)
```

---

n_distinct *Number of Unique Values in an* ore.vector *Object*

Description

Counts the number of unique values in an ore.vector object.

Usage

```
n_distinct(x, na_rm = FALSE)
```

Arguments

x	An ore.vector object.
na_rm	Specifies whether to include NA value.

Examples

```
x <- sample(1:10, 1e5, rep = TRUE)
X <- ore.push(x)
length(unique(X))
n_distinct(X)
```

---

ore-boxplot.stats *Box Plot Statistics*

Description

Calculates statistics necessary for producing box plots.

Usage

```
  ## S4 method for signature 'ore.vector'
boxplot.stats(x, coef = 1.5, do.conf = TRUE,
              do.out = TRUE)
```

Arguments

x	An ore.number object.
coef	A numeric value that specifying the maximum relative distance the whiskers can extend out from the box.
do.conf,do.out	Logical values indicating if the conf or out elements of the list output are NULL values.

Value

Returns a list conforming to the output of boxplot.stats.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

boxplot.stats

---

ore-class *Class* ore

Description

The ore virtual class that sits at the top of the Oracle R Enterprise class hierarchy.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector

Examples

```
showClass("ore")
```

ore-coplot *Conditioning Plots*

Description

Creates conditioning plots.

Usage

```
  ## S4 method for signature 'ANY,ore.frame'
coplot(formula, data, given.values,
         panel = points, rows, columns, show.given = TRUE,
         col = par("fg"), pch = par("pch"),
         bar.bg = c(num = gray(0.8), fac = gray(0.95)),
         xlab, ylab, subscripts = FALSE,
         axlabels = function(f) abbreviate(levels(f)),
         number = 6, overlap = 0.5, xlim, ylim, ...)
```

Arguments

formula	A formula describing the form of conditioning plot.
data	An ore.frame object.
given.values, panel, rows, columns, show.given, col, pch, bar.bg, xlab, ylab, subscrip	See the descriptions of these arguments in coplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

coplot

ore-data-image-class *Class* ore-data-image

Description

A class containing R object(s) and PNG images returned from Embedded R Execution when graphical output is captured.

List Methods

x$dat: R object returned from embedded R execution.

x$img: A list of PNG images returned from embedded R execution.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

```
library(png)
res <- ore.tableApply(ore.push(iris[,1, drop = FALSE]),
               function(dat, n) { plot(seq(n))
                                  n + length(dat)
               },
               n = 10, ore.graphics = TRUE)
class(res)
res1 <- ore.pull(res, graphics = TRUE)
class(res1)
res1$dat
dim(readPNG(res1$img[[1L]], native = TRUE))
```

---

ore-matplot *Plot Columns of Matrices*

Description

Plots the columns of one matrix against the columns of another.

Usage

```
  ## S4 method for signature 'ore'
matplot(x, y, type = "p", lty = 1:5, lwd = 1,
          lend = par("lend"), pch = NULL, col = 1:6, cex = NULL,
          bg = NA, xlab = NULL, ylab = NULL, xlim = NULL, ylim = NULL,
          ..., add = FALSE, verbose = getOption("verbose"))
  ## S4 method for signature 'ore'
matlines(x, y, type = "l", lty = 1:5, lwd = 1,
          pch = NULL, col = 1:6, ...)
  ## S4 method for signature 'ore'
matpoints(x, y, type = "p", lty = 1:5, lwd = 1,
          pch = NULL, col = 1:6, ...)
```

Arguments

x,y	ore objects.
type,lty,lwd,lend,pch,col,cex,bg,xlab,ylab,xlim,ylim,...,add,verbose	See description in matplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

matplot

---

ore-polygon *Polygon Drawing*

Description

Draws polygons.

Usage

```
  ## S4 method for signature 'ore'
polygon(x, y = NULL, density = NULL, angle = 45,
          border = NULL, col = NA, lty = par("lty"), ...,
          fillOddEven = FALSE)
```

Arguments

x,y	ore.number objects.
density,angle,border,col,lty,...,fillOddEven	See description in polygon.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

polygon

ore-polypath *Path Drawing*

Description

Draws paths.

Usage

```
  ## S4 method for signature 'ore'
polypath(x, y = NULL, border = NULL, col = NA,
           lty = par("lty"), rule = "winding", ...)
```

Arguments

x,y	ore.number objects.
border,col,lty,rule,...	See description in polypath.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

polypath

ore-rug *Add a Rug to a Plot*

Description

Draws rug ticks.

Usage

```
  ## S4 method for signature 'ore.vector'
rug(x, ticksize = 0.03, side = 1, lwd = 0.5,
      col = par("fg"),quiet = getOption("warn") < 0, ...)
```

Arguments

x	ore.number objects.
ticksize,side,lwd,col,quiet,...	See description in rug.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

rug

---

ore-segments *Add Line Segments to a Plot*

Description

Draws line segments between pairs of points.

Usage

```
  ## S4 method for signature 'ore.vector'
segments(x0, y0, x1 = x0, y1 = y0,
           col = par("fg"), lty = par("lty"), lwd = par("lwd"), ...)
```

Arguments

x0,y0,x1,y1	ore.number objects.
col,lty,lwd,...	See description in segments.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

segments

ore-smoothScatter *Scatterplots with Smoothed Densities Color Representation*

Description

Creates a color kernel density smoothed representation of the scatterplot.

Usage

```
  ## S4 method for signature 'ore'
smoothScatter(x, y = NULL, nbin = 128, bandwidth,
              colramp = colorRampPalette(c("white", blues9)),
              nrpoints = 100, ret.selection = FALSE,
              pch = ".", cex = 1, col = "black",
              transformation = function(x) x^.25,
              postPlotHook = box, xlab = NULL, ylab = NULL,
              xlim, ylim, xaxs = par("xaxs"), yaxs = par("yaxs"),
              ...)
```

Arguments

x,y	ore.number objects.
nbin,bandwidth,colramp,nrpoints,ret.selection,pch,cex,col,transformation,postPlo	See description in smoothScatter.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

smoothScatter

ore-symbols *Draw Symbols (Circles, Squares, Stars, Thermometers, Boxplots)*

Description

Draws symbols on a plot.

Usage

```
  ## S4 method for signature 'ore.vector'
symbols(x, y = NULL, circles, squares,
          rectangles, stars, thermometers, boxplots, inches = TRUE,
          add = FALSE, fg = par("col"), bg = NA,
          xlab = NULL, ylab = NULL, main = NULL,
          xlim = NULL, ylim = NULL, ...)
```

Arguments

x,y	ore.number objects.
circles,squares,rectangles,stars,thermometers,boxplots,inches,add,fg,bg,xlab,yla	See description in symbols.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

symbols

---

ore-xspline *Draw and X-spline*

Description

Draws an X-spline, which is a curve drawn relative to control points.

Usage

```
  ## S4 method for signature 'ore'
xspline(x, y = NULL, shape = 0, open = TRUE,
          repEnds = TRUE, draw = TRUE, border = par("fg"), col = NA,
          ...)
```

Arguments

x,y	ore.number objects.
shape,open,repEnds,draw,border,col,...	See description in xspline.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

xspline

ore-xy.coords *Extract Plotting Structures*

Description

Obtains x and y coordinates for plotting.

Usage

```
  ## S4 method for signature 'ore'
xy.coords(x, y = NULL, xlab = NULL, ylab = NULL,
            log = NULL, recycle = FALSE, ...)
```

Arguments

x,y	ore.number objects.
xlab,ylab,log,recycle	See description in xy.coords.
...	Additional arguments for future extension.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

xy.coords

ore.attach *Oracle R Enterprise Environment Management Functions*

Description

Attaches or detaches the R environment of the named schema to or from the R search path in the Oracle R Enterprise session. When the R environment is attached to the R search path, the corresponding ore.frame objects, representing database tables and views, in the schema can be accessed directly by name.

Usage

```
ore.attach(schema, pos = 2, warn.conflicts = TRUE)
ore.detach(schema)
```

Arguments

schema	A character string specifying the database schema name.
pos	An integer value larger than 1 specifying the position in the R search path.
warn.conflicts	A logical value indicating whether or not warnings should be printed for conflicts, objects that share the same name, that result from attaching the R environment.

Details

If argument schema is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

By default the R environment for a schema in the Oracle R Enterprise session is attached in position 2 in the R search path, immediately after the user's workspace and before all previously attached packages and environment. The pos argument can be used to attach the R environment for a schema in the Oracle R Enterprise session at a different location in the search path, but it cannot be attached at pos = 1.

The functions ore.sync and ore.rm add and remove ore objects from this R environment for a schema in the Oracle R Enterprise session.

Value

For the ore.attach function, the environment is returned invisibly with a "name" attribute.

For the ore.detach function, the return value is an invisible NULL value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.connect, ore.exists, ore.get, ore.ls, ore.rm, ore.sync, search

Examples

```
if (!interactive())
{
  search()
  ore.attach()
  search()
  ore.detach()
  search()
  ore.attach("rquser", 3)
  search()
  ore.detach("rquser")
  search()
}
```

ore.character-class
Class ore.character

Description

The ore.character class represents character data columns in Oracle R Enterprise.

Character Data Methods

- casefold(x,upper = FALSE): Returns an ore.character object containing the upper case or lower case form of argument x.
- chartr(old,new,x): Returns an ore.character object containing a character translation version of argument x.
- grepl(pattern,x): Returns an ore.logical object indicating whether or not the specified pattern argument is found in the elements of argument x.
- gsub(pattern,replacement,x): Returns an ore.character object containing a version of argument x where all matches to argument pattern are replaced with argument replacement.
- nchar(x,type = "chars"): Returns an ore.integer object containing the number of characters or bytes for the elements in argument x.
- paste(...,sep = " "): Returns an ore.character object containing a concatenation of the values in arguments ....
- sub(pattern,replacement,x): Returns an ore.character object containing a version of argument x where the first match to argument pattern is replaced with argument replacement.
- substr(x,start,stop): Returns an ore.character object containing a the specified substrings for the elements of argument x that begin at the location of argument start and end at the location of argument stop.
- substring(text,first,last = 1000000L): Returns an ore.character object containing a the specified substrings for the elements of argument text that begin at the location of argument first and end at the location of argument last.
- tolower(x): Returns an ore.character object containing the lower case form of argument x.
- toupper(x): Returns an ore.character object containing the upper case form of argument x.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector

Examples

```
x <- ore.push("MiXeD cAsE 123")
class(x)
chartr("iXs", "why", x)
chartr("a-cX", "D-Fw", x)
tolower(x)
toupper(x)
```

---

ore.connect *Oracle R Enterprise Connection Functions*

Description

Establishes or terminates a connection to an Oracle R Enterprise, Apache Hive or Apache Impala server.

Usage

```
ore.is.connected(type = c("ORACLE", "HIVE", "IMPALA"))
ore.connect(user = "", sid = "", host = "localhost", password = "",
            port = NULL, service_name = NULL, conn_string = NULL,
            all = FALSE, type = c("ORACLE", "HIVE", "IMPALA"),
            tzone = Sys.getenv("TZ"), schema = NULL, ...)
ore.disconnect()
```

Arguments

user	If argument type = "ORACLE" (the default), then a character string specifying the Oracle Database user name. If argument type = "HIVE" or type = "IMPALA", then a character string specifying the Apache Hive or Apache Impala user name.
sid	When argument type = "ORACLE" (the default), a character string specifying the Oracle Database SID or Oracle Wallet connection string. If type = "HIVE" or type = "IMPALA", then this argument is ignored.
host	If argument type = "ORACLE" (the default), then a character string specifying the host name of the Oracle Database server. If type = "HIVE" or type = "IMPALA", then a character string specifying the host name of the Apache Hive or Apache Impala.
password	If argument type = "ORACLE" (the default), then a character string specifying the Oracle Database user password. If type = "HIVE" or type = "IMPALA", then the connection attempt uses the Apache Hive or Apache Impala user name.
port	If argument type = "ORACLE" (the default), then a number specifying the Oracle Database port (usually 1521). If type = "HIVE" or type = "IMPALA", then a number specifying the Apache Hive or Apache Impala port.

service_name When argument type = "ORACLE" (the default), a character string specifying the service name to be used in the connection identifier for the Oracle Database instance. If type = "HIVE" or type = "IMPALA", then this argument is ignored.

conn_string When argument type = "ORACLE" (the default), a character string specifying the connection string used to connect to the Oracle Database instance. Usually used for a database connection with Oracle Wallet. If type = "HIVE" or type = "IMPALA", then this parameter is ignored.

all A logical value indicating whether to call functions ore.sync and ore.attach using their default arguments. This results in synchronizing and attaching all tables and views in the schema for the Oracle R Enterprise connection. The execution time for ore.sync grows linearly with the number of visible tables and views.

type A character string specifying the database type: either "ORACLE" (default) to connect to an Oracle Database instance or "HIVE" to connect to an Apache Hive server or type = "IMPALA" to connect to an Apache Impala server.

tzone A character string specifying the session time zone for both the R and database sessions. The default value of tzone is the value of the system environment variable TZ. If the value of tzone is either NA or an empty string, then UTC is used for both the R and database sessions. For type = "HIVE" or type = "IMPALA", this parameter is ignored.

schema A character string specifying the schema. For type = "HIVE" or type = "IMPALA", this parameter specifies the target database.

... Additional parameters for a Hive or Impala connection when running Hiveserver2 in different modes such as:

- Hive Specific Configuration parameters: Use the parameter hiveconf to specify the list of Apache Hive Server configuration parameters to set for Hive connection.
- HiveServer2 with Kerberos authentication: Use the parameter principal to specify the Kerberos server principal for the host where HiveServer2 is running.
- Hiveserver2 with SSL: In this mode, use parameters such as ssl="true", sslTrustStore to specify the path to the client's truststore file and trustStorePassword to specify the password for the truststore.
- Impala with Kerberos authentication: In this mode, use parameters such as AuthMech="1" to indicate Kerberos authentication. Use KrbRealm="realm.example.c and KrbHostFQDN="Kerberos_host_name" to specify the realm and Kerberos Host FQDN (Fully Qualified Domain Name). Use KrbServiceName="impala to specify the Kerberos service name for Impala.
- Any other configuration of HiveServer2: There are many such configuration combinations in which HiveServer2 could be running. Check the Hive Documentation for the various modes and parameters needed to connect in those modes.

Details

Functions ore.connect and ore.disconnect are called for their side effects, namely establishing or terminating a connection to an Oracle R Enterprise, Apache Hive or Apache Impala server. The call to function ore.connect must precede all other calls to Oracle R Enterprise, Apache Hive Apache Impala functionality (except ore.is.connected). There can only be one active

Oracle R Enterprise, Apache Hive or Apache Impala connection. An Oracle R Enterprise, Apache Hive or Apache Impala session can optionally end with a call to function `ore.disconnect`. An Oracle R Enterprise, Apache Hive or Apache Impala session is implicitly terminated when the R session ends.

Both the R and the database session time zones are set during the execution of function `ore.connect`. The session time zones are set to the value of the argument `tzone`, whose default value is equal to the system environment variable `TZ`. If both `TZ` and `tzone` are not specified, the session time zones are set to `UTC`.

If you are using Oracle Wallet to store username and password, then use the `conn_string` argument to pass the connection string for the wallet mapping (for more information refer to chapter 3 "Configuring Clients to Use the External Password Store" of Oracle Database Security Guide). In addition, the `service_name` argument can be used to pass service name information. (Refer to the chapter 2 of Oracle Database Net Services Administrator's Guide for more information on service names).

If you are connecting to Hiveserver2 with Kerberos authentication, then use the parameter `principal` to specify the Kerberos server principal for the host where Hiveserver2 is running (See the example below).

If you are connecting to Impala with Kerberos authentication, then you must use parameters specific to Impala. Do not use the Hive with Kerberos parameters. (See the example below).

If Apache Sentry is enabled on a cluster, then it is fully supported and compatible with Apache Hive. All the conditions and access restrictions imposed by Apache Sentry on Hive tables are applied when accessing it through the ORE transaparency layer.

Calling `ore.connect` during an active Oracle R Enterprise connection of any type disconnects that session before starting a new session.

Value

A logical value in the case of function `ore.is.connected` indicating whether an active connection to an Oracle Database instance or Apache Hive or Apache Impala server exists; otherwise an invisible `NULL` value.

Author(s)

Oracle `<oracle-r-enterprise@oracle.com>`

References

Oracle R Enterprise

See Also

`ore.attach`, `ore.exists`, `ore.get`, `ore.ls`, `ore.rm`, `ore.sync`

Examples

```
## Not run:
  # pass the username password during connect
  if (!is.ore.connected())
    ore.connect("rquser", "orcl", "localhost", "rquser")
  ore.ls()
  ore.disconnect()
```

```
  # pass the connect string for wallet mode
  if (!is.ore.connected())
    ore.connect(conn_string = "<wallet_connect_string>")
  ore.ls()
  ore.disconnect()

  # Apache Hive Connection
  if (!is.ore.connected(type="HIVE"))
    ore.connect(host="localhost", port=10000, user="user", password="password", schema="d
  ore.ls()
  ore.disconnect()

  # Apache Impala Connection
  if (!is.ore.connected(type="IMPALA"))
    ore.connect(host="localhost", port=21500, user="user", password="password", schema="d
  ore.ls()
  ore.disconnect()

  # Apache Hive with Kerberos authentication
  if (!is.ore.connected(type="HIVE"))
    ore.connect(user="user", password="password", host="localhost", port=10000, type="HIV
  ore.ls()
  ore.disconnect()

  # Apache Impala with Kerberos authentication
  if (!is.ore.connected(type="IMPALA"))
    ore.connect(host="localhost", port=21050, AuthMech="1", KrbRealm="DEV.EXAMPLE.COM", K
  ore.ls()
  ore.disconnect()

## End(Not run)
```

`ore.const` *Oracle R Enterprise Constants*

Description

Holds mathematical constants for SQL calculations.

Usage

```
ore.const
```

Format

An `environment` object containing:

`"1_PI" 1/pi`

`"1_SQRT2PI" 1/sqrt(pi)`

`"2PI" 2*pi`

`"2_PI" 2/pi`

`"2_SQRTPI" 2/sqrt(pi)`

`"E" exp(1)`

"LN10" log(10)
"LN2" log(2)
"LNSQRT2PI" log(sqrt(2*pi))
"LOG10E" log10(e)
"LOG2E" log2(e)
"PI" pi
"PI_2" pi/2
"PI_4" pi/4
"SQRT1_2" 1/sqrt(2)
"SQRT2" sqrt(2)

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

```
ore.const
ore.const$PI
ore.const$"2_SQRTPI"
```

---

ore.corr *Oracle R Enterprise Correlation Analysis*

Description

Performs correlation analysis across numeric columns in ore.frame objects. Supports partial correlations with a control column specification and allows the specification of aggregations prior to computing correlations.

Usage

```
ore.corr(data, var, stats = "pearson", group.by = NULL, freq = NULL,
         with = NULL, weight = NULL, partial = NULL)
```

Arguments

data	An ore.frame object.
var	A comma-separated character string specifying the names of numeric columns within argument data.
stats	A character string specifying the correlation type; one of "pearson" (default), "spearman" or "kendall".
group.by	An optional character vector specifying the group by column names within argument data.

freq	An optional character string specifying a numeric column within argument data to use as a frequency count. If a frequency value is less than 1 or missing, the observation is excluded from correlation calculation. If a frequency value is not an integer, it is truncated.
with	An optional character vector specifying the numeric columns in argument data to pair with columns specified in argument var. For example, with var = c('x1','x2') and with = c('y1','y2','y3'), function ore.corr will compute the following correlation pairs: (x1,y1), (x1,y2), (x1,y3), (x2,y1), (x2,y2) and (x2,y3).
weight	An optional character string specifying a numeric column within argument data to use as analytic weights. Cannot be used with stats = "kendall".
partial	An optional character vector specifying the numeric columns within argument data to use as control variables for partial correlations.

Value

When argument group.by is not specified, returns an ore.frame object.

When argument group.by is specified, returns a list of ore.frame objects.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

cor

Examples

```
# Copy iris data set to the database
IRIS <- ore.push(iris)

# Pearson's correlation
x <- cor(iris[,1:3])
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length")

# Kendall's tau
x <- cor(iris[,1:3], method = "kendall")
y <-  ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
               stats = "kendall")

# Partial correlation
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
              partial = "Petal.Width")

# Group by partial correlation
y <- ore.corr(IRIS, var = "Sepal.Length, Sepal.Width, Petal.Length",
              partial = "Petal.Width", group.by = "Species")
```

ore.create *Database Table Creation or Destruction Functions*

Description

Create an Oracle Database table from a data.frame or ore.frame object, or a view from an ore.frame object.

Drop a database table or view.

Usage

```
ore.create(x, table = NULL, view = NULL, append = FALSE, ...)
ore.drop(table = NULL, view = NULL, model = NULL, silent = FALSE)
```

Arguments

x	A ore.frame object. Can also be a data.frame if argument table is used.
table	A character string specifying the name of the table. This argument cannot be used with argument view or model.
view	A character string specifying the name of the view. This argument cannot be used with argument table or model.
model	A character string specifying the name of the model. This argument cannot be used with argument table or view.
silent	A boolean (TRUE or FALSE) indicating whether to suppress warning message in case object does not exists.
append	A boolean value currently supported in Hive and Impala only, if set to TRUE, data.frame/ore.frame can be appended to an existing table. Default value is FALSE.
...	Additional parameters for a Hive connection when using Hiveserver2 in different modes such as: • To Overwrite the current data in table: Use the parameter overwrite to specify whether to overwrite the data in the table supported in Hive and Impala. • type of connection: Use parameter type to specify the connection type. A character string specifying to create an INTERNAL/EXTERNAL table. Defaults to internal table

Details

Exactly one of arguments table or view must be specified, depending on whether a database table or view is desired for creation or dropping.

When a new object is created using function ore.create, it is automatically included into the list of objects that can be accessed via ore.get and ore.ls functions. It will also be added to the list of attached ore.frame objects if ore.attach was already called. Also for function ore.create, only an ore.frame object can be used with the view argument.

Similarly, when an object is removed using function ore.drop, it will no longer show up in output of ore.ls function. It will also be removed from a list of attached ore.frame objects.

If the input object contains column names that do not match the naming convention of the backend server they will be modified by `ore.make.names` function.

Function `ore.create` does not preserve `row.names` neither in the database table nor in the newly created `ore.frame` object.

Value

Both functions `ore.create` and `ore.drop` return an invisible `NULL` value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.ls`, `ore.exists`, `ore.sync`, `ore.rm`, `ore.attach`, `ore.connect`

Examples

```
if (!interactive())
{
  ore.drop(table = "TEN_LETTERS", silent = TRUE)
  ore.drop(table = "IRIS_TABLE", silent = TRUE)
  ore.drop(view = "IRIS_VIEW", silent = TRUE)

  ore.create(data.frame(x = 1:10, y = letters[1:10]),
             table = "TEN_LETTERS")
  ore.exists("TEN_LETTERS")

  ore.create(iris, table = "IRIS_TABLE")
  ore.create(head(ore.get("IRIS_TABLE"), 10), view = "IRIS_VIEW")

  ore.exists("IRIS_VIEW")

  ore.drop(table = "TEN_LETTERS")
  ore.drop(table = "IRIS_TABLE")
  ore.drop(view = "IRIS_VIEW")

  X <- ore.push(iris)
  em1 <- OREdm::ore.odmEM(~., X, odm.settings = list(model_name = "emmod"))
  ore.drop(model = "emmod")
  ore.drop(model = "emmod")
  ore.drop(model = "emmod", silent = TRUE)
}
```

ore.crosstab *Oracle R Enterprise Crosstabulations*

Description

Expands on function xtabs by supporting multiple columns with optional aggregations, weighting, and ordering options. Building cross-tabulation is a pre-requisite to using the function ore.freq.

Usage

```
ore.crosstab(expr, data, ..., group.by = NULL, order = NULL,
             weights = NULL, where = NULL, strata = NULL)
```

Arguments

expr	A formula object defining a cross-tabulation. Syntax: [CS] ~ CS [*<WC>][/<GC>][^[SC][|O
	where CS is [+CSET][+CRANGE], CSET is [+CSET], and CRANGE is <FROM COLUMN>-<TO COLUMN>.
data	An ore.frame object.
...	Additional arguments.
group.by	An optional character vector specifying the group by column names within argument data.
order	An optional character string specifying an ordering for the cross-tabulation; one of "NAME" (ascending by name), "-NAME" (descending by name), "DATA" (ascending by data), "-DATA" (descending by data), "FREQ" (ascending by frequency), "-FREQ" (descending by frequency), "INTERNAL".
weights	An optional character string specifying a numeric column within argument data to use as analytic weights.
where	An optional character vector specifying arbitrary partitions of argument data.
strata	An optional character string specifying a column within argument data to use as stratification variable.

Value

When argument where is not specified, returns an ore.frame object.

When argument where is specified, returns a list of ore.frame objects.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.freq, xtabs

Examples

```
IRIS <- ore.push(iris)

table(iris$Species, iris$Petal.Length)
ore.crosstab(Species ~ Petal.Length, data = IRIS)

# 1 way table
ore.crosstab(~Species, data = IRIS)

# 2 2-way tables
ore.crosstab(Species ~ Petal.Length + Sepal.Length, data = IRIS)

# Order rows of cross-tabulation by asc frequency counts
ore.crosstab(Species ~ Petal.Length | FREQ, data = IRIS)

# Order by descending frequency counts
ore.crosstab(Species ~ Petal.Length | -FREQ, data = IRIS)

# As many cross-tabs as distinct values in Species
ore.crosstab(Petal.Length ~ Sepal.Length / Species, data = IRIS)

# Syntactic simplication
ore.crosstab(Sepal.Length - Petal.Width ~ Species, data = IRIS)

# Illustration of the use of group.by
ore.crosstab(Species ~ Petal.Length, data = IRIS)

# Compare with the following where Petal.Length values are hidden
ore.crosstab(~ Species, group.by = "Petal.Length", data = IRIS)

# Use of derived columns
IRIS$PetalBins <- ifelse(IRIS$Petal.Length < 2, "SMALL",
                  ifelse(IRIS$Petal.Length < 5, "MEDIUM", "LARGE"))
ore.crosstab(Species ~ PetalBins, data = IRIS)
```

---

ore.datastore *Oracle R Enterprise Datastore Listing Function*

Description

Lists existing datastores in the user's Oracle Database schema.

Usage

```
ore.datastore(name, pattern, type = c("user", "private", "all",
                                      "grantable", "grant", "granted"))
```

Arguments

name	An optional character string specifying the datastore name; cannot be used with argument pattern.
pattern	An optional regular expression character string specifying the matching datastore names; cannot be used with argument name.

type — An optional scalar character string specifying the type of datastore to list. The valid values are 'user' (default), 'private', 'all', 'grantable', 'grant', or 'granted'. 'user' lists the datastores created by current session user. 'private' lists these datastores the read privileges for which cannot be granted by the current session user to other users. 'all' lists all the datastores to which the current session user has read access. 'grantable' lists these datastores the read privilege for which can be granted by the current session user to other users. 'grant' lists these datastores the read privilege for which has been granted by the current session user to other users. 'granted' lists these datastores the read privilege for which has been granted by other users to the current session user.

Details

Function `ore.datastore` lists high-level information about datastores in the user's Oracle Database schema. This datastore listing may be optionally filtered by either the `name` or `pattern` argument.

Value

A `data.frame` object is returned with columns `datastore.name`, `object.count`, `size`, `creation.date`, and `description` when argument `type` is 'user', 'private', or 'grantable'; these columns specify the datastore name, the number of objects in the named datastore, the size of the datastore in bytes, the datastore creation date, and the datastore comment, respectively. The output `data.frame` has an extra column `owner` specifying the owner of the datastores when argument `type` is 'all' or 'granted'. When argument `type` is 'grant', the output `data.frame` contains columns `datastore.name` and `grantee` specifying the datastore name and the name of the user to which the read privilege of the named datastore has been granted by the current session user. The rows of the `data.frame` object are alphabetically sorted according to column `owner`, if present, and `datastore.name`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.save`, `ore.load`, `ore.move`, `ore.delete`, `ore.datastoreSummary` `ore.grant` `ore.revoke`

Examples

```
if (!interactive())
{
if (any(sapply(c("x", "y", "z"), exists)))
  stop("object x, y, or z exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)
```

```
ore.save(list=ls(), name="rqds_1")
ore.save(x, y, name="rqds_2")
ore.save(z, name="rqds_3", grantable=TRUE)

# list overall information about the datastores in user's schema
ds <- ore.datastore()
## Not run:
ds

## End(Not run)


# list overall information about the datastores available in the user's schema
ds <- ore.datastore(type="all")
## Not run:
ds

## End(Not run)


# list overall information about the datastore the read privilege for which
# can be granted
ds <- ore.datastore(type="grantable")
## Not run:
ds

## End(Not run)


# list overall information about the datastore with name 'rqds_1'
ds <- ore.datastore(name="rqds_1")
## Not run:
ds

## End(Not run)


# list overall information about the datastores whose name starts
# with 'rqds_'
ds <- ore.datastore(pattern="^rqds_")
## Not run:
ds

## End(Not run)


sapply(c("rqds_1", "rqds_2", "rqds_3"), ore.delete)
rm(x, y, z)
}
```

`ore.datastoreSummary`

Oracle R Enterprise Datastore Summarizing Function

Description

Summarizes the contents of a datastore in the user's Oracle Database schema.

Usage

```
ore.datastoreSummary(name, owner)
```

Arguments

name	The character string specifying the datastore name.
owner	An optional character string specifying the owner of the datastore to summarize. If the owner of the datastore is not the current user, the read privilege for the datastore must be granted to the current user by ore.grant; otherwise ore.load errors out. If the owner argument not specified, the default value is the current user.

Details

Summarizes the datastore specified by argument name and owner. If the named datastore does not exist or the current user does not have read privilege for this datastore, the function throws an error.

Value

A data.frame object is returned with columns object.name, class,size, length, row.count, and col.count; these columns specify the object name, the object's class, the object's size in bytes, the length of object, and the number of rows and columns of object, respectively. The rows of the data.frame object are alphabetically sorted by column object.name.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.move, ore.delete, ore.datastore ore.grant ore.revoke

Examples

```
## Not run:
if (any(sapply(c("x", "y", "z"), exists)))
  stop("object x, y, or z exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

ore.save(list=ls(), name="rq$ds_1")

# show detailed information about the datastore rq$ds_1
```

```
ore.datastoreSummary(name="rq$ds_1")

ore.delete(name="rq$ds_1")
rm(x, y, z)

## End(Not run)
```

`ore.datetime-class` *Class* `ore.datetime`

Description

The `ore.date` and `ore.datetime`, and `ore.difftime` classes represent `DATE`, `TIMESTAMP`, and `INTERVAL DAY TO SECOND` data columns in Oracle R Enterprise.

Details

`ore.datetime` objects, unlike `POSIXt` objects, do not support a time zone setting. Instead all `ore.datetime` use the system time zone, `Sys.timezone()`, if is available or GMT if this information is not available.

Date/Time Arithmetic

The following arithmetic operations are supported for date/time data types:

Arith Op	Signature	Value
"+"	ore.date, ore.difftime	ore.date
"+"	ore.difftime,ore.date	ore.date
"+"	ore.date, ore.number	ore.date
"+"	ore.number, ore.date	ore.date
"+"	ore.datetime,ore.difftime	ore.datetime
"+"	ore.difftime,ore.datetime	ore.datetime
"+"	ore.datetime,ore.number	ore.date
"+"	ore.number, ore.datetime	ore.date
"+"	ore.difftime,ore.difftime	ore.difftime
"−"	ore.date, ore.date	ore.difftime
"−"	ore.date,ore.datetime	ore.difftime
"−"	ore.datetime,ore.date	ore.difftime
"−"	ore.date, ore.difftime	ore.date
"−"	ore.date, ore.number	ore.date
"−"	ore.datetime,ore.datetime	ore.difftime
"−"	ore.datetime,ore.difftime	ore.datetime
"−"	ore.datetime, ore.number	ore.date
"−"	ore.difftime,ore.difftime	ore.difftime
"*"	ore.difftime,ore.number	ore.difftime
"*"	ore.number,ore.difftime	ore.difftime
"/"	ore.difftime,ore.number	ore.difftime

In addition to binary arithmetic operators, date/time data support:

`diff(x,lag = 1,differences = 1)`: Returns an `ore.difftime` object containing a suit-

ably lagged and iterated differences of argument x.

lag An integer indicating which lag to use.

differences An integer indicating the order of the difference.

trunc(x,units = "DD",...): Returns a ore.datetime object containing a truncated version of argument x.

x An ore.datetime object.

units A character string specifying the level of truncation, one of

units	Description
CC, SCC	Century.
D	Day of week (1-7).
DAY	Name of day.
DD	Day of month (1-31).
DDD	Day of year (1-366).
DY	Abbreviated name of day.
HH, HH12	Hour of day (1-12).
HH24	Hour of day (0-23).
IW	Week of year (1-52 or 1-53) based on the ISO standard.
IYY, IY, I	Last 3, 2, or 1 digit(s) of ISO year.
IYYY	4-digit year based on the ISO standard.
J	Julian day; the number of days since January 1, 4712 BC.
MI	Minute (0-59).
MM	Month (01-12; January = 01).
MON	Abbreviated name of month.
MONTH	Name of month.
Q	Quarter of year (1, 2, 3, 4; January - March = 1).
RM	Roman numeral month (I-XII; January = I).
WW	Week of year (1-53) where week 1 starts on the first day of the year and continues to the seventh day of t
W	Week of month (1-5) where week 1 starts on the first day of the month and ends on the seventh.
Y, YYY	Year with comma in this position.
YEAR, SYEAR	Year, spelled out; S prefixes BC dates with a minus sign (-).
YYYY, SYYYY	4-digit year; S prefixes BC dates with a minus sign.
YYY, YY, Y	Last 3, 2, or 1 digit(s) of year.

Character Coersion

as.character(x,format = "YYYY-MM-DD",...) and as.ore.character(x,format = "YYYY-MM-DD",...) for ore.date; as.character(x,format = "YYYY-MM-DD HH24:MI:SS",...) and as.ore.character(x,format = "YYYY-MM-DD HH24:MI:SS",...) for ore.datetime: Returns an ore.character object containing information from argument x in a form specified by the format argument.

format	Description
-, /, ,, ., ;, :, "text"	Punctuation and quoted text is reproduced in the result.
AD, A.D.	AD indicator with or without periods.
AM, A.M.	Meridian indicator with or without periods.
BC, B.C.	BC indicator with or without periods.
CC, SCC	Century.
D	Day of week (1-7).
DAY	Name of day.
DD	Day of month (1-31).

DDD	Day of year (1-366).
DL	Returns a value in the long date format.
DS	Returns a value in the short date format.
DY	Abbreviated name of day.
E	Abbreviated era name.
EE	Full era name.
FF [1..9]	Fractional seconds.
FM	Returns a value with no leading or trailing blanks.
FX	Requires exact matching between the character data and the format model.
HH, HH12	Hour of day (1-12).
HH24	Hour of day (0-23).
IW	Week of year (1-52 or 1-53) based on the ISO standard.
IYY, IY, I	Last 3, 2, or 1 digit(s) of ISO year.
IYYY	4-digit year based on the ISO standard.
J	Julian day; the number of days since January 1, 4712 BC.
MI	Minute (0-59).
MM	Month (01-12; January = 01).
MON	Abbreviated name of month.
MONTH	Name of month.
PM, P.M.	Meridian indicator with or without periods.
Q	Quarter of year (1, 2, 3, 4; January - March = 1).
RM	Roman numeral month (I-XII; January = I).
RR	Lets you store 20th century dates in the 21st century using only two digits.
RRRR	Round year.
SS	Second (0-59).
SSSSS	Seconds past midnight (0-86399).
TS	Returns a value in the short time format.
TZD	Daylight saving information.
TZH	Time zone hour.
TZM	Time zone minute.
TZR	Time zone region information.
WW	Week of year (1-53) where week 1 starts on the first day of the year and continues to the seven
W	Week of month (1-5) where week 1 starts on the first day of the month and ends on the sevent
X	Local radix character.
Y, YYY	Year with comma in this position.
YEAR, SYEAR	Year, spelled out; S prefixes BC dates with a minus sign (-).
YYYY, SYYYY	4-digit year; S prefixes BC dates with a minus sign.
YYY, YY, Y	Last 3, 2, or 1 digit(s) of year.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.year, ore, ore.character, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector.

Examples

```
DATE <- ore.push(seq(as.Date("1999/1/1"), as.Date("2002/1/1"), by="3 months"))
DATE
as.character(DATE, format = "YYYY/MM/DD")
```

---

ore.delete *Oracle R Enterprise Datastore Removal Function*

Description

Deletes one or more datastores from the user's Oracle Database schema or specific R objects from within a datastore.

Usage

```
ore.delete(name, list = character(0), pattern)
```

Arguments

name	A character string specifying the name of the datastore to delete or modify; cannot be used with argument pattern.
list	An optional character vector containing the names of the objects to delete from the datastore specified by name. If this argument is not specified, then the entire datastore is deleted.
pattern	Optional. Cannot be used with either argument name or list. Can be either a single unnamed regular expression character string or a named vector of them. If it is a single regular expression character string, then the argument specifies the names of datastores to delete. If it is a vector of regular expression character strings, then each element of the vector specifies the names of objects to delete from the datastore specified by the name of the element. If an element is NA, then the entire datastore is deleted. If the name of an element is NA, then an error is thrown.

Details

If argument list is missing, and argument name is used, then function ore.delete deletes the datastore specified in the name argument from the user's Oracle Database schema.

If arguments name and list are used, then function ore.delete deletes the specified R objects from the datastore supplied in the name argument.

If argument pattern is a single character string, then function ore.delete deletes the datastores in the user's Oracle Database schema whose names match the regular expression specified in the pattern argument.

If argument pattern is a named vector of character strings, then for each element in the list, function ore.delete deletes from the datastore specified by the name of the element those R objects whose names match the regular expression of the element.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.load, ore.move, ore.datastore, ore.datastoreSummary

Examples

```
if (any(sapply(c("rqobj1", "rqobj2", "y", "z"), exists)))
  stop("object rqobj1, rqobj2, y, or z exists")

rqobj1 <- matrix(c(TRUE, FALSE, TRUE, TRUE, TRUE, FALSE), nrow = 2, ncol = 3)
rqobj2 <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(rqobj2)

ore.save(rqobj1, rqobj2, y, z, name="rq$ds_1")
ore.save(y, name="rq$ds_2a")
ore.save(z, name="rq$ds_2b")
ore.save(rqobj1, rqobj2, y, name="rq$ds_3")
ore.save(y, z, name="rq$ds_4")

# delete R object y, z from datastore 'rq$ds_1' from user's schema
ore.delete(name="rq$ds_1", list=c("y", "z"))
# delete the datastore with name 'rq$ds_1' from user's schema
ore.delete(name="rq$ds_1")
# delete datastores with names starting with 'rq$ds_2' from user's schema
sort(ore.delete(pattern="^rq\$ds_2"))

# delete R objects whose names contain 'rqobj' from datastore 'rq$ds_3' from user's schem
ore.delete(pattern=c("rq$ds_3"="rqobj"))

# delete the contents of datastore 'rq$ds_4' from user's schema, but not the datastore it
ore.delete(pattern=c("rq$ds_4"=".*"))

# delete datastores 'rq$ds_3' and 'rq$ds_4' from user's schema
ore.delete(pattern=c("rq$ds_3"=NA, "rq$ds_4"=NA))
# delete all datastores
## Not run:
ore.delete(pattern=".*")

## End(Not run)
rm(rqobj1, rqobj2, y, z)
```

---

ore.doEval *Oracle R Enterprise Embedded R Script Execution Functions*

Description

Runs a function within the Oracle database under various conditions.

Usage

```
ore.doEval(FUN, ..., FUN.VALUE = NULL, FUN.NAME = NULL, FUN.OWNER = NULL)
ore.groupApply(X, INDEX, FUN, ..., FUN.VALUE = NULL,
               FUN.NAME = NULL, FUN.OWNER = NULL,
               parallel = getOption("ore.parallel", NULL))
ore.indexApply(times, FUN, ..., FUN.VALUE = NULL,
               FUN.NAME = NULL, FUN.OWNER = NULL,
               parallel = getOption("ore.parallel", NULL))
ore.rowApply(X, FUN, ..., FUN.VALUE = NULL,
             FUN.NAME = NULL, FUN.OWNER = NULL, rows = 1,
             parallel = getOption("ore.parallel", NULL))
ore.tableApply(X, FUN, ..., FUN.VALUE = NULL,
               FUN.NAME = NULL, FUN.OWNER = NULL)
```

Arguments

X	An ore.frame object.
INDEX	A ore.vector or ore.frame object containing ore.factor objects or columns, each of which is the same length as argument X. It is used to partition the data in X before sending it to function FUN. The counterpart supported R types are logical, integer, numeric, character, factor.
times	The number of times to execute the function.
FUN	The function to be applied. For functions ore.groupApply, ore.rowApply, and ore.tableApply the first argument to the FUN argument must represent a data.frame object. For function ore.indexApply, the first argument to FUN must represent the index number. For function ore.doEval, no arguments are required for FUN. The function specified by FUN cannot recursively call embedded R APIs. Cannot be used with argument FUN.NAME.
...	Additional arguments to FUN. Arguments that start with ore. are special control arguments. They are not passed to the function specified by FUN or FUN.NAME arguments, but instead control what happens before or after the execution of the closure. The following control arguments are supported: 1. ore.drop - controls the object type for the input data. If TRUE, a one column data.frame will be converted to a vector. The default value is TRUE. 2. ore.na.omit - controls the handling of missing values in the input data. If TRUE, rows or vector elements, depending on the ore.drop setting, containing missing values will be removed from the input data. If all the rows in a chunk contain missing values, the input data for that chunk will be an empty data.frame or vector. The default value is FALSE. 3. ore.connect - controls whether to automatically connect to Oracle R Enterprise inside the closure. This is equivalent to doing an ore.connect call with the same credentials as the client session. The default value is FALSE. 4. ore.graphics - controls whether to start a graphical driver and look for images. The default value is TRUE. 5. ore.png.* - if ore.graphics is TRUE, additional parameters for the png graphics device driver. The naming convention for these arguments is to add an ore.png. prefix to the arguments of the png function. For

example, if ore.png.height is supplied, argument height will be passed to the png function. If not set, the standard default values for the png function are used.

6. ore.envAsEmptyenv - controls whether referenced environments in an object should be replaced with an empty environment during serialization. Some types of input parameters and returned objects, such as list, formula, are serialized before being saved to the database. If TRUE, the referenced environment in the object will be replaced with an empty environment whose parent is .GlobalEnv, and therefore, the objects in the original referenced environment will not be serialized. In some situations, this could significantly reduce the size of serialized objects. If FALSE, all the objects in the referenced environment will be serialized, and could be unserialized and recovered later. The default value is regulated by the global option ore.envAsEmptyenv.
7. ore.characterAsFactor - controls the type that character and factor columns of the in-database data set referred to by X are treated as. If TRUE, all character and factor columns are treated as factor type. If FALSE, all character and factor columns are treated as character type. For functions ore.groupApply and ore.rowApply, each partition will be only aware of factor levels in itself and will not be aware of levels in other partitions. The default value is FALSE.

FUN.VALUE

A data.frame or ore.frame to use as a template for the return value.

The attribute ora.type can be applied to a data.frame column to specify that the corresponding output column of a ore.frame uses a CLOB or BLOB type.

FUN.NAME

A character string specifying the name of a serialized R script, which contains a single R function definition, within the Oracle R Enterprise in-database R script archive. Cannot be used with FUN.

Oracle R Enterprise comes with a number of predefined graphical scripts. All predefined scripts have a reserved name that start with RQG$ followed by a function name from the **graphics** package that the script wraps. Depending on the function it either takes the first, the first and second or all columns of the input data.frame. Thus, predefined scripts can only be used with ore.tableApply, ore.groupApply, or ore.rowApply. Each function also has a ... so that it can pass any parameter to the function that it wraps. Here is a list of predefined graphical scripts:

1. RQG$plot1d - a wrapper for plot. Works on the first column of the input data.frame object.
2. RQG$plot2d - a wrapper for plot. Works on the first two columns of the input data.frame object.
3. RQG$hist - a wrapper for hist. Works on the first column of the input data.frame object.
4. RQG$boxplot - a wrapper for boxplot. Works on the first column of the input data.frame object.
5. RQG$smoothScatter - a wrapper for smoothScatter. Works on the first two columns of the input data.frame object.
6. RQG$cdplot - a wrapper for cdplot. Works on the first two columns of the input data.frame object.
7. RQG$pairs - a wrapper for pairs. Works on all columns of the input data.frame object.

8. RQG$matplot - a wrapper for matplot. Works on all columns of the input data.frame object.

Oracle R Enterprise also comes with a number of predefined R and package version scripts. These scripts start with RQ$ followed by an R function name that the script wraps and can only be used with ore.doEval. Here is a list of these predefined scripts:

1. RQ$R.Version - a wrapper for R.Version. Takes no argument and returns R version-relevant information.
2. RQ$getRversion - a wrapper for getRversion. Takes no argument and returns R version number.
3. RQ$installed.packages - a wrapper for installed.packages. Takes no argument and returns package name, version number, and package installation location of installed packages.
4. RQ$packageVersion - a wrapper for packageVersion. Takes package name as argument and returns package version number.

FUN.OWNER — An optional character string specifying the owner of the FUN.NAME R script. The user that creates an R script with ore.scriptCreate is the owner of that script. The RQSYS schema is the owner of the global and pre-defined R scripts. When FUN.OWNER is not specified or is NULL, then Oracle R Enterprise looks for the owner in the following order: user of the current session, RQSYS. Argument FUN.OWNER is only used with argument FUN.NAME.

rows — The maximum number of rows in each chunk.

parallel — A preferred degree of parallelism to use in the embedded R job; either a positive integer greater than or equal to 2 for a specific degree of parallelism, a value of FALSE or 1 for no parallelism, a value of TRUE for the data argument's default parallelism, or NULL for the database default for the operation. The default value is regulated by the global option ore.parallel.

Details

Function ore.doEval executes a function, either FUN or FUN.NAME, within an R process running inside the Oracle database.

Function ore.groupApply partitions an in-database data set by a (potentially derived) column and executes a function on those partitions within R processes running inside the Oracle database. Each partition must fit wholly within a single R process.

Function ore.indexApply executes a function index number of times inside the Oracle database.

Function ore.rowApply partitions an in-database data set into row chunks and executes a function on those partitions within R processes running inside the Oracle database. Each partition must fit wholly within a single R process.

Function ore.tableApply executes a function on an in-database data set.

Either argument FUN or FUN.NAME must be supplied. For security reasons, use of argument FUN requires 'RQADMIN' Oracle database privileges. Because creation of the R script represented by argument FUN.NAME has to be published by someone with 'RQADMIN' credentials, it can be used by anyone authorized to use Oracle R Enterprise.

Argument FUN.OWNER can be used with argument FUN.NAME to uniquely specify an R function defined in the R script repository.

The parallel argument regulates the use of a '/*+ parallel */', '/*+ parallel(DOP) */', or a '/*+ no_parallel */' hint being added to the underlying SQL query. Consult Oracle database documentation for more information.

The function to be applied specified via argument FUN or FUN.NAME is automatically connected to Oracle R Enterprise with the same credentials as the client session invoking it. Only an equivalent of ore.connect is invoked. Functions such as ore.sync, ore.attach, and ore.get should be called explicitly.

Value

If argument FUN.VALUE is supplied, an ore.frame object that conforms to the FUN.VALUE template is returned.

If argument FUN.VALUE is not supplied, then functions ore.doEval and ore.tableApply return an ore.object while functions ore.groupApply, ore.indexApply, and ore.rowApply return an ore.list.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.scriptCreate, ore.options

Examples

```
## ore.doEval
eval1 <- ore.doEval(function() "Hello, world")
eval2 <-
  ore.doEval(function()
             data.frame(x = "Hello, world", stringsAsFactors = FALSE))
eval3 <-
  ore.doEval(function()
             data.frame(x = "Hello, world", stringsAsFactors = FALSE),
             FUN.VALUE =
             data.frame(x = character(), stringsAsFactors = FALSE))
out.df <- data.frame(x = character(), y = raw(), stringsAsFactors = FALSE)
attr(out.df$x, "ora.type") <- "clob"
attr(out.df$y, "ora.type") <- "blob"
eval4 <-
  ore.doEval(function() {
             res <- data.frame(x = "Hello, world",stringsAsFactors = FALSE)
             res$y[[1L]] <- charToRaw("Hello, world")
             res},
             FUN.VALUE = out.df)
eval1
class(eval1)  # ore.object
eval2
class(eval2)  # ore.object
eval3
class(eval3)  # ore.frame
eval4$x
rawToChar(ore.pull(eval4$y))

## copy data to the database
```

```
IRIS <- ore.push(iris)

## ore.groupApply
grpAp1 <-
  ore.groupApply(IRIS, IRIS$Species,
                 function(df)
                 if(nrow(df) == 0)
                     NULL
                 else
                     summary(lm(Sepal.Length ~ .,  data = df[1:4])),
                 parallel = TRUE)
grpAp2 <-
  ore.groupApply(IRIS, IRIS$Species,
                 function(df) {
                     if (nrow(df) == 0) {
                         species <- character()
                         cf <- numeric()
                         names(cf) <- character()
                     } else {
                         species <- as.character(df$Species[1])
                         cf <- coef(lm(Sepal.Length ~ .,
                                       data = df[1:4]))
                     }
                     data.frame(Species = species,
                                CoefName = names(cf),
                                CoefValue = unname(cf),
                                stringsAsFactors = FALSE)
                 },
                 FUN.VALUE =
                 data.frame(Species = character(),
                            CoefName = character(),
                            CoefValue = numeric(),
                            stringsAsFactors = FALSE),
                 parallel = TRUE)
class(grpAp1)  # ore.list
class(grpAp2)  # ore.frame

## ore.indexApply
ore.indexApply(5, function(i) i)
if (interactive())
    ore.indexApply(5, function(i) summary(rnorm(100)), parallel = TRUE)

## ore.rowApply
# create a classification tree for iris data
library(rpart)
irisRpart <- rpart(Species ~ ., data = iris)

irisPred <-
  ore.rowApply(IRIS,
               function(df, model) {
                   library(rpart)
                   cbind(df, PRED = predict(model, df, type = "class"))
               }, model = irisRpart,
               FUN.VALUE =
               cbind(iris[integer(),], PRED = character()),
               rows = 50, parallel = TRUE)
```

```
## ore.tableApply
ore.tableApply(IRIS, function(df) summary(df))
```

`ore.esm` *Oracle R Enterprise Time Series Exponential Smoothing Models*

Description

Creates exponential smoothing models on ordered `ore.vector` data.

Usage

```
ore.esm(x,
        interval = NULL,
        model = "simple",
        accumulate = "NONE",
        setmissing = "PREV",
        optim.start = c(alpha=0.3, beta=0.1),
        optim.control = list())

## S3 method for class 'ore.esm'
fitted(object, start = NULL, end = NULL, ...)
## S3 method for class 'ore.esm'
predict(object, n.ahead = 12L,  ...)

forecast.ore.esm(object, h = 12L, ...)
```

Arguments

`x`	An ordered `ore.vector` of time series data or transactional data. The ordering column could be either integers from 1 to the length of the time series or of type `ore.datetime`.
`interval`	The interval of the time series, or the time interval by which the transactional data is to be accumulated. If the ordering column of the argument `x` is of type `ore.datetime`, `interval` must be specified. Possible values: `"YEAR"`, `"QTR"`, `"MONTH"`, `"WEEK"`, `"DAY"`, `"HOUR"`, `"MINUTE"`, `"SECOND"`
`model`	The exponential smoothing model name. Possible values: `"simple"`, `"double"`
`accumulate`	The method of accumulation. Possible values:

`"NONE"`	No accumulation occurs. In this case, the argument `x` is required to be equally spaced time series observa
`"TOTAL"`	Accumulation based on the sum of the observed values.
`"AVERAGE"`	Accumulation based on the average of the observed values. The value could be abbreviated to `"AVG"`.
`"MINIMUM"`	Accumulation based on the minimum of the observed values. The value could be abbreviated to `"MIN"`.
`"MAXIMUM"`	Accumulation based on the maximum of the observed values. The value could be abbreviated to `"MAX"`.
`"MEDIAN"`	Accumulation based on the median of the observed values. The value could be abbreviated to `"MED"`.
`"STDDEV"`	Accumulation based on the standard deviation of the observed values. The value could be abbreviated to
`"N"`	Accumulation based on the number of nonmissing observations.
`"NOBS"`	Accumulation based on the number of observations.
`"NMISS"`	Accumulation based on the number of missing observations.

`setmissing` The method of treating missing values.
Possible values:

`"AVERAGE"`	Missing values are set to the average of the accumulated values. The value could be abbreviated to `"AV`
`"MINIMUM"`	Missing values are set to the minimum of the accumulated values. The value could be abbreviated to `"M`
`"MAXIMUM"`	Missing values are set to the maximum of the accumulated values. The value could be abbreviated to `"M`
`"MEDIAN"`	Missing values are set to the median of the accumulated values. The value could be abbreviated to `"ME`
`"FIRST"`	Missing values are set to the first accumulated nonmissing value.
`"LAST"`	Missing values are set to the last accumulated nonmissing value.
`"PREVIOUS"`	Missing values are set to the previous accumulated nonmissing value. The value could be abbreviated to
`"NEXT"`	Missing values are set to the next accmulated nonmissing value.

`optim.start` A vector with named components `alpha` and `beta` containing the starting values for the optimizer. The starting values should be in the range of 0 to 1. Ignored in the `simple` model case.

`optim.control`
Optional list with additional control parameters passed to `optim` in the `double` model case. Ignored in the `simple` model case.

`object` An object of type `ore.esm`.

`start` A positive integer that specifies the beginning index of the output of the fitted values. If a value is specified, it should be less than or equal to the argument `end`. The default is `NULL`, which indicates `start=1`.

`end` A positive integer that specifies the ending index of the output of the fitted values. If a value is specified, it should be greater than or equal to the argument `start`. The default is `NULL`, which indicates the value is equal to the length of the training dataset.

`n.ahead` The number of time periods to forecast.

`h` The number of time periods to forecast.

`...` Additional arguments.

Details

The function `ore.esm` implements exponential smoothing models for in-database time series observations. The function can work with either time series data, whose observations are evenly spaced by a fixed interval, or transactional data, whose observations are not equally spaced. For transactional data, specify an aggregation method with the argument `accumulate` and a time interval with the argument `interval`. To handle missing values, specify a value with the argument `setmissing`. The function preprocesses the data using the specified aggregation method, time intervals, and handling of missing values.

The `fitted` method provides the fitted times series values aligned with the training dataset. The arguments `start` and `end` specify the index range of the output fitted values. This function currently does not support a model built with an accumulation method specified.

The `predict` method predicts the time series by using the exponential smoothing model built by `ore.esm`. If the `forecast` package is loaded, the `forecast` method can be called through generic.

Value

For `ore.esm`, returns an object of class `"ore.esm"`. Some of its components are as follows:

`smoothing.param`
: The estimated smoothing parameters.

`model`
: The model type.

`model.param`
: The input arguments.

For `fitted.ore.esm`, returns an `ore.vector` of fitted values of the model in the range that is specified by the arguments `start` and `end`.

For `predict.ore.esm` or `forecast.ore.esm`, returns a `data.frame` of the predicted time series.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Exponential Smoothing method http://en.wikipedia.org/wiki/Exponential_smoothing

Oracle R Enterprise

Examples

```
# case 1
N <- 5000
ts0 <- ore.push(data.frame(ID=1:N,
  VAL=seq(1,5,length.out=N)^2+rnorm(N,sd=0.5)))
rownames(ts0) <- ts0$ID
x <- ts0$VAL

esm.mod <- ore.esm(x, model = "double")
esm.predict <- predict(esm.mod, 30)
esm.fitted <- fitted(esm.mod, start=4000, end=5000)
plot(ts0[4000:5000,], pch='.')
lines(ts0[4000:5000, 1], esm.fitted, col="blue")
lines(esm.predict, col="red", lwd=2)

#case 2
ts01 <- data.frame(ID=seq(as.POSIXct("2008/6/13"), as.POSIXct("2011/6/16"),
                   length.out=4000), VAL=rnorm(4000, 10))
ts02 <- data.frame(ID=seq(as.POSIXct("2011/7/19"), as.POSIXct("2012/11/20"),
                   length.out=1500), VAL=rnorm(1500, 10))
ts03 <- data.frame(ID=seq(as.POSIXct("2012/12/09"), as.POSIXct("2013/9/25"),
                   length.out=1000), VAL=rnorm(1000, 10))
ts1 = ore.push(rbind(ts01, ts02, ts03))
rownames(ts1) <- ts1$ID
x <- ts1$VAL
esm.mod <- ore.esm(x, "DAY", accumulate = "AVG", model="simple",
                   setmissing="PREV")
esm.predict <- predict(esm.mod)

#case 3
x <- ore.push(BJsales)

esm.mod <- ore.esm(x, model="double")
esm.predict <- predict(esm.mod)
esm.fitted <- fitted(esm.mod)
```

```
library(OREgraphics)
plot(x)
lines(esm.fitted, col="blue")
lines(esm.predict, col="red", lwd=4)
```

`ore.exec` *SQL Query Execution Function*

Description

Executes the specified SQL query in the user's Oracle Database schema.

Usage

```
ore.exec(qry)
```

Arguments

`qry`	A character string specifying the SQL statement.

Details

The `ore.exec` function is intended for use with data definition language (DDL) type statements, such as CREATE TABLE, that have no return value.

If the `ore.exec` function call creates a table in the user's Oracle Database schema, a corresponding `ore.frame` object is not created unless the user invokes function `ore.sync`. To avoid refreshing all `ore.frame` objects, the name of the table can be explicitly provided.

Value

Returns an invisible `NULL` value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.ls`, `ore.sync`, `dbGetQuery`

Examples

```
if (!interactive())
{
  ore.exec("create table TEST_CREATE as select * from dual")
  ore.exists("TEST_CREATE")      # table not found
  ore.sync(table="TEST_CREATE")  # sync metadata of new table only
  ore.exists("TEST_CREATE")      # table TEST_CREATE is present
  ore.drop(table="TEST_CREATE")  # clean up
  ore.exists("TEST_CREATE")
}
```

---

ore.exists *Oracle R Enterprise Object Existence Checking Function*

Description

Determines whether a named ore.frame object, representing a database table or view, exists within the R environment for a schema in the Oracle R Enterprise session.

Usage

```
ore.exists(name, schema)
```

Arguments

name	A character string specifying the name of the ore.frame object.
schema	A character string specifying the database schema name.

Details

If argument schema is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

Value

A logical value indicating whether an ore.frame object of the given name is found within the specified schema.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.attach, ore.connect, ore.get, ore.ls, ore.rm, ore.sync

Examples

```
if (!interactive())
{
  ore.exists("IRIS_TABLE")                # FALSE
  ore.create(iris, table = "IRIS_TABLE")
  ore.exists("IRIS_TABLE")                # TRUE
  ore.exists("IRIS_TABLE", "RQUSER")      # TRUE
  ore.drop(table = "IRIS_TABLE")
}
```

---

ore.factor-class *Class* ore.factor

Description

The ore.factor class represents factor data columns in Oracle R Enterprise.

Details

The levels of ore.factor objects are not managed as a lookup table the same way that factor objects are. Instead the levels for ore.factor objects are equivalent to sort(unique(as.character(x))).

Factor Data Methods

interaction(...,sep = "."): Returns an ore.factor object containing the combination of the values in arguments ....

levels(x): Returns a character vector containing the factor levels.

nlevels(x): Returns an integer value containing the number of factor levels.

summary(object,maxsum = 100,...): Returns a tabular summary of the ore.factor object.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.datetime, ore.character, ore.frame, ore.logical, ore.matrix, ore.number, ore.vector

Examples

```
sdiv <- ore.push(state.division)
class(sdiv)               # ore.factor
levels(sdiv)
nlevels(sdiv)
summary(sdiv)

sdiv2 <- ore.pull(sdiv)
class(sdiv2)              # factor
```

---

ore.frame-class *Class* ore.frame

Description

The ore.frame class represents data sets in Oracle R Enterprise.

Accessors

In the code snippets below, argument x is an ore.frame object.

nrow(x), ncol(x): Returns the number of rows and columns, respectively.

NROW(x), NCOL(x): Same as nrow(x) and ncol(x), respectively.

dim(x): Returns an integer vector defined as c(nrow(x),ncol(x)).

length(x): Returns the number of columns.

colnames(x), names(x): Returns a character vector containing the column names.

colnames(x) <-value, names(x) <-value: Replaces the column names in argument x with the names in argument value.

rownames(x), row.names(x): Returns an ore.character object containing the row names. When the row names are made of multiple components, they will be separated with the value specified in the ore.sep option, which by default is set to "|".

rownames(x) <-value, row.names(x) <-value: Replaces the row names in argument x with the names in argument value. The value argument must be either NULL to remove row names, an ore.vector object for single component names, or an ore.frame object for multiple component names.

dimnames(x): Returns a list object defined as list(NULL,colnames(x)).

Subsetting

In the code snippets below, argument x is an ore.frame object. The vector selecting columns can be logical, numeric or character. The behavior is similar to that of data.frame with a few exceptions or rather extensions. When doing character subsetting on an ore.frame with complex rownames only the first component is used for subsetting. Subsetting on all components is not yet supported.

x[i,j,drop=TRUE]: Returns a new ore.frame object made of the selected rows and columns. For single column selection, the drop argument specifies whether or not to return an ore.vector object.

x[j] <-value, x[,j] <-value: Replaces the specified columns in x with value. When value is NULL, the specified columns are removed from x.

`x[[i]]`, `x$name`: Returns the selected column as an `ore.vector` object.

`x[[i]] <-value`, `x$name <-value`: Replaces the specified column in `x` with `value`. When `value` is `NULL`, the column is removed from `x`.

`head(x,n = 6L)`: If argument `n` is non-negative, returns the first `n` rows of argument `x`. If argument `n` is negative, returns all but the last `abs(n)` rows of argument `x`.

`tail(x,n = 6L)`: If argument `n` is non-negative, returns the last `n` rows of argument `x`. If `n` is negative, returns all but the first `abs(n)` rows of argument `x`.

`subset(x,subset,select,drop = FALSE)`: Returns a new `ore.frame` object using:

subset A logical expression indicating rows to keep, where missing values are taken as a logical `FALSE` value.

select An expression indicating columns to keep.

drop Argument passed on to `[` indexing operator.

`is.na(x)`: Returns an `ore.frame` object containing `logical` data columns that indicate which cells contains missing values.

`is.finite(x)`: Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain finite numbers.

`is.infinite(x)`: Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain infinite values.

`is.nan(x)`: Returns an `ore.frame` object containing `logical` data columns that indicate which cells contain a not-a-number value.

Splitting and Combining

In the code snippets below, argument `x` is an `ore.frame` object.

`split(x,f,drop = FALSE)`: Splits argument `x` into a `list` object, according to argument `f`, dropping elements corresponding to unrepresented levels if `drop` is `TRUE`.

`cbind(...)`: Returns a new `ore.frame` object by combining the columns of the `ore.frame` objects in `...`.

`rbind(...)`: Returns a new `ore.frame` object by combining the rows of the `ore.frame` objects in `...`.

`merge(x,y,...)`: Merges two `ore.frame` objects `x` and `y`, with arguments in `...` being the same as those allowed by the base `merge` function. It is allowed for either arguments `x` or `y` to be a `data.frame` object. If both inputs are ordered `ore.frame` objects, the result will also be an ordered `ore.frame` object with a complex key made from `x` and `y` keys. Otherwise, the result is unordered.

Looping

In the code snippets below, argument `x` is an `ore.frame` object.

`by(data,INDICES,FUN,...,simplify = TRUE)`: Apply argument `FUN` to each partitioning of argument `data`, an `ore.frame` object, specified by the `factor` (or `list` of `factor` objects) argument `INDICES`.

Utilities

In the code snippets below, argument `x` is an `ore.frame` object.

`unique(x)`: Returns a new `ore.frame` object that contains only the distinct rows in argument `x`.

Evaluation

`eval(expr,envir,enclos = parent.frame())`: Converts the `ore.frame` object specified in argument `envir` to an `environment` using function `as.env`, with argument `enclos` as its parent, and then evaluates argument `expr` within that environment.

`with(data,expr,...)`: Equivalent to expression `eval(quote(expr),data,...)`.

`within(data,expr,...)`: Similar to function `with`, except assignments made during evaluation are taken as assignments into argument `data`, i.e., new symbols have their value appended to argument `data`, and assigning new values to existing symbols results in replacement.

`transform(`_data`,...)`: Similar to function `within`, except assignments are specified as independent optional arguments instead as depended evaluations within a single expression.

Matrix Methods

`colMeans(x,na.rm = FALSE)`: Returns an `ore.number` object containing the column means of argument `x`.

`colSums(x,na.rm = FALSE)`: Returns an `ore.number` object containing the column sums of argument `x`.

`rowMeans(x,na.rm = FALSE)`: Returns an `ore.number` object containing the row means of argument `x`.

`rowSums(x,na.rm = FALSE)`: Returns an `ore.number` object containing the row sums of argument `x`.

`scale(x,center = TRUE,scale = TRUE)`: Returns an `ore.frame` object containing the possibly centered and scaled version of argument `x`.

`max.col(m,ties.method = c("first","last"))`: Returns an `ore.integer` object containing the column number of the maximum value for each row of argument `m`.

Group Generics

`ore.frame` objects have support for S4 group generic functionality:

```
Arith "+", "-", "*", "^", "%%", "%/%", "/"
Compare "==", ">", "<", "!=", "<=", ">="
Logic "&", "|"
Ops "Arith", "Compare", "Logic"
Math "abs", "sign", "sqrt", "ceiling", "floor", "trunc", "cummax", "cummin",
    "cumprod", "cumsum", "log", "log10", "log2", "log1p", "acos", "acosh",
    "asin", "asinh", "atan", "atanh", "exp", "expm1", "cos", "cosh", "sin",
    "sinh", "tan", "tanh", "gamma", "lgamma", "digamma", "trigamma"
Math2 "round", "signif"
Summary "max", "min", "range", "prod", "sum", "any", "all"
```

See S4groupGeneric for more details.

Logical Methods

`!x`: Returns an `ore.frame` object containing the logical negation (NOT) of argument `x`.

`xor(x,y)`: Returns an `ore.frame` object containing the exclusive or combination of arguments `x` and `y`.

Coercion

In the code snippets below, argument x is an ore.frame object.

as.env(x,enclos = parent.frame()): Returns an environment object containing an ore.vector for each column in argument x.

as.list(x): Returns a list object containing an ore.vector for each column in argument x.

as.matrix(x): Returns an ore.matrix object.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.logical, ore.matrix, ore.number, ore.vector

Examples

```
IRIS <- ore.push(iris)
head(IRIS)
summary(IRIS)
colMeans(IRIS[1:4])
```

---

ore.frame-factanal *Oracle R Enterprise Factor Analysis*

Description

Factor analysis of ore.frame data.

Usage

```
  ## S4 method for signature 'ANY'
factanal(x, factors, data = NULL, covmat = NULL,
           n.obs = NA, subset, na.action, start = NULL,
           scores = c("none", "regression", "Bartlett"),
           rotation = "varimax", control = NULL, ...)
```

Arguments

x	A formula or ore.frame object containing numeric columns.
factors	The number of factors in the analysis.
data	An optional ore.frame to use when argument x is a formula.
covmat, n.obs	Ignored when supplying ore.frame input data.
subset	An optional subset expression.
na.action	The manner in which to handle NA values, either na.omit or na.pass.
start	Optional starting values.
scores	The type of scores to produce; only settings of "none" or "regression" are supported.
rotation	The type of rotation to use.
control	A list of control values.
...	Additional optional arguments.

Value

A factanal object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

factanal, ore.frame-prcomp, ore.frame-princomp

Examples

```
LONGLEY <- ore.push(longley)

factanal(LONGLEY, 3)
factanal(~ GNP + Unemployed + Population + Employed, 1, data = LONGLEY)
```

---

ore.frame-model.frame

Oracle R Enterprise Modeling Framework

Description

Model functions related to variable preparation of ore.frame data in statistical modeling.

Usage

```
  ## S4 method for signature 'ore'
complete.cases(...)
  ## S4 method for signature 'formula,ore.frame'
get_all_vars(formula, data = NULL, ...)
  ## S4 method for signature 'formula'
model.frame(formula, data = NULL, subset = NULL,
      na.action = getOption("na.action", "na.omit"),
      drop.unused.levels = FALSE, xlev = NULL, ...)
  ## S4 method for signature 'formula'
model.matrix(object, data = environment(object),
      contrasts.arg = NULL, xlev = NULL,
      na.action = getOption("na.action", "na.omit"), ...)
  ## S4 method for signature 'ore.frame'
na.omit(object, ...)
```

Arguments

formula	A model formula or a terms object.
object	For the model.matrix method, a model formula or a terms object. For the na.omit method, an ore.frame object.
data	An ore.frame object.
subset	An expression object for row selection.
na.action	The manner in which NA values are handled, either na.omit or na.pass.
drop.unused.levels	This argument is not supported.
xlev	A named list of character vectors specifying the levels for each ore.factor variable.
contrasts.arg	A named list of contrasts, defined as numeric matrices or character strings specifying contrast functions, for the ore.factor variables.
...	Additional arguments.

Value

For the complete.cases method, returns an ore.logical object.

For the get_all_vars, model.frame, and na.omit methods, returns an ore.frame object.

For the model.matrix method, returns an ore.matrix object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.getXlevels, complete.cases, get_all_vars, model.frame, model.matrix, na.action

Examples

```
X <- ore.push(data.frame(V1 = c(-1:2, NA),
                         V2 = c(1:4, NA),
                         V3 = c(NA, rep(c("a", "b"), 2)),
                         V4 = c(NA, rep(c("A", "B"), c(2, 2)))))
complete.cases(X)
get_all_vars(V1 ~ log(V2) * V3, data = X)
model.frame(V1 ~ log(V2) * V3, data = X)
model.matrix(V1 ~ log(V2) * V3, data = X)
na.omit(X)
```

---

ore.frame-prcomp *Oracle R Enterprise Principal Components Analysis*

Description

Principal components analysis of ore.frame data.

Usage

```
  ## S4 method for signature 'ANY'
prcomp(x, ...)
  # prcomp(formula, data = NULL, subset, na.action, ...)
  # prcomp(x, retx = TRUE, center = TRUE, scale. = FALSE,
  #        tol = NULL, ...)
```

Arguments

x	See Details section below.
...	Additional arguments.

Details

This is a wrapper method around the prcomp function in the **stats** package to support the analysis of ore.frame objects. As with the original function, it can be used through two different signatures: prcomp(formula, data = NULL, subset, na.action, ...) and prcomp(x, retx = TRUE, center = TRUE, scale. = FALSE, tol = NULL, ...). Descriptions of the arguments for these two function signatures are the following:

formula A formula with no response variable that refers only to numeric variables.

data An ore.frame object that contains the variables specified in the formula argument.

subset An optional subset expression.

na.action The manner in which to handle NA values, either na.omit or na.pass.

x An ore.frame object that contains only numeric data.

retx A logical value indicating whether the rotated variables should be returned.

center The center argument to be used internally by the scale function.

scale. The scale argument to be used internally by the scale function.

tol See the description of the tol argument in prcomp.

Value

A prcomp object.

Note

The function biplot currently does not work on the prcomp object returned by this ORE wrapper method.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

prcomp, ore.frame-factanal, ore.frame-princomp

Examples

```
USARRESTS <- ore.push(USArrests)

prcomp(USARRESTS)
prcomp(USARRESTS, scale. = TRUE)

# forumla interface
prcomp(~ Murder + Assault + UrbanPop, data = USARRESTS, scale. = TRUE)
```

---

ore.frame-princomp *Oracle R Enterprise Principal Components Analysis*

Description

Performs principal components analysis of ore.frame data.

Usage

```
  ## S4 method for signature 'ANY'
princomp(x, ...)
  # princomp(formula, data, subset, na.action, cor = FALSE,
  #          scores = TRUE, ...)
  # princomp(x, subset, na.action, cor = FALSE, scores = TRUE, ...)
```

Arguments

x	See Details section below.
...	Additional arguments.

Details

This is a wrapper method around the princomp function in the **stats** package to support the analysis of ore.frame objects. As with the original function, it can be used through two different signatures: princomp(formula,data,subset,na.action,cor = FALSE,scores = TRUE,...) and princomp(x,subset,na.action,cor = FALSE,scores = TRUE,...). Descriptions of the arguments for these two function signatures are the following:

formula A formula with no response variable that refers only to numeric variables.

data An ore.frame object that contains the variables specified in the formula argument.

subset An optional subset expression.

na.action The manner in which to handle NA values, either na.omit or na.pass.

x An ore.frame object that contains only numeric data.

cor A logical value that indicates whether the principal components should be based on the correlation matrix (cor = TRUE) or the covariance matrix (cor = FALSE).

scores A logical value that indicates whether the principal component scores should be included in the output.

Value

A princomp object.

Note

The function biplot currently does not work on the princomp object returned by this ORE wrapper method.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

princomp, ore.frame-factanal, ore.frame-prcomp

Examples

```
USARRESTS <- ore.push(USArrests)

princomp(USARRESTS)
princomp(USARRESTS, cor = TRUE)

# forumla interface
princomp(~ Murder + Assault + UrbanPop, data = USARRESTS, cor = TRUE)
```

`ore.freq` *Oracle R Enterprise Frequency Analysis*

Description

Operates on output from the function `ore.crosstab` and automatically determines techniques that are relevant for the table. For 1 way cross tables, goodness of fit tests for equal proportions or specific null proportions, confidence limits and tests are equivalence are supported. For 2 way cross tables, various statistics that describe relationships between columns involved in the cross tabulation are supported. This includes chi-square tests, cochran-mantel-haenzsel statistics, measures of association, strength of association, risk differences, odds ratio and relative risk for 2x2 tables and tests for trend. N-way tables specifications are build as N, 2 way cross tables.

Usage

```
ore.freq(x, stats = NULL, params = NULL, persist = FALSE,
         skip.failed = FALSE, skip.missing = FALSE, use.ext = FALSE,
         use.r = FALSE, use.sql = FALSE)
```

Arguments

`x`	Output from function `ore.crosstab`.
`stats`	A comma-separated character string specifying one or more of the following methods to use on argument `x`: Chi-Square Tests : `"AJCHI"`, `"LRCHI"`, `"MHCHI"`, `"PCHISQ"` Kappa Tests: `"KAPPA"`, `"WTKAP"` Lambda Tests: `"LAMCR"`, `"LAMRC"`, `"LAMDAS"` Correlation: `"KENTB"`, `"PCORR"`, `"SCORR"` Stuart's Tau, Somer's D|C: `"STUTC"`, `"SMDCR"`, `"SMDRC"` Fisher's Test: `"FISHER"` Cochran's Q: `"COCHQ"` Odds Ratio: `"OR"`, `"MHOR"`, `"LGOR"` Relative Risk: `"RR"`, `"MHRR"`, `"ALRR"` Cramer's V: `"CRAMV"` Gamma: `"GAMMA"` Trend Test: `"TREND"` McNemar's Test: `"MCNEM"`
`params`	A character string specifying the control parameters for the methods in argument `stats`. Possible setting include: Scoring Options: `"SCORE=TABLE|RANK|RIDIT|MODRIDIT"` Alpha criterion: `"ALPHA=<Number>"` Weights: `"WEIGHTS=<Number>"`
`persist`	For internal use only.
`skip.failed`	A logical value indicating whether to return with an error if a particular statistical test fails or to skip over the in-applicable test.
`skip.missing`	A logical value indicating whether to skip cells with missing values in argument `x`.
`use.ext, use.r, use.sql`	For internal use only.

Value

Returns an `ore.frame` object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.crosstab

Examples

```
## Not run:
  IRIS <- ore.push(iris)
  x <- ore.crosstab(Species ~ Petal.Length + Sepal.Length, data = IRIS)
  ore.freq(x, stats = "SCORR, GAMMA, STUTC")

## End(Not run)
```

---

ore.get *Oracle R Enterprise Object Retrieval Function*

Description

Retrieves the specified ore.frame object, representing a database table or view, from the R environment for a schema in the Oracle R Enterprise session.

Usage

```
ore.get(name, schema)
```

Arguments

name	A character string specifying the name of the ore.frame object.
schema	A character string specifying the database schema name.

Details

If argument schema is unspecified, a searches is conducted across the attached R environments for the schemas in the Oracle R Enterprise session, according to the search list order, and returns the first object instance found.

Value

If found within argument schema, returns an ore.frame object. If not found, an error is produced.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.attach, ore.connect, ore.exists, ore.ls, ore.rm, ore.sync

Examples

```
if (!interactive())
{
  ore.create(data.frame(x=3, y="A", z=TRUE), table = "TABLE_GET")
  # search in the default schema
  ore.get("TABLE_GET")
  # search in the specified schema
  ore.get("TABLE_GET", "RQUSER")
  ore.drop(table = "TABLE_GET")
}
```

---

ore.getXlevels *Factor Levels for an Oracle R Enterprise Model Matrix*

Description

Functions that either create a list of factor levels that can be used in the xlev argument of a model.matrix call involving an ore.frame object or a table containing the number of factor levels in such a list object.

Usage

```
ore.getXlevels(Terms, m)
ore.getXnlevels(Terms, m)
```

Arguments

Terms	A terms or formula object.
m	An ore.frame or data.frame object that was generated by a model.frame function call.

Details

The ore.getXlevels function is the Oracle R Enterprise equivalent to the .getXlevels function in the **stats** package.

Value

For ore.getXlevels, a named list containing the factor levels for the categorical variables derived in the Terms argument.

For ore.getXnlevels, a table containing the number of factor levels in each of the categorical variables derived in the Terms argument.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`.getXlevels`, ore.frame-model.frame

Examples

```
X <- ore.push(data.frame(V1 = -1:2,
                         V2 = 1:4,
                         V3 = rep(c("a", "b"), 2),
                         V4 = rep(c("A", "B"), c(2, 2))))
trms <- terms(V1 ~ log(V2) * V3 * V4)
mf   <- model.frame(trms, data = X)
ore.getXlevels(trms, mf)
ore.getXnlevels(trms, mf)
```

---

ore.grant — *Oracle R Enterprise Privilege Grant and Revoke Functions*

Description

Grants or revokes read privilege for an R script or datastore.

Usage

```
ore.grant(name, type = c("datastore", "rqscript"), user = NULL)
ore.revoke(name, type = c("datastore", "rqscript"), user = NULL)
```

Arguments

`name`	A character string that specifies the name of an R script in the R script repository or the name of a datastore. The current user must be the owner of the R script or datastore.
`type`	A scalar character string specifying either 'datastore' or 'rqscript' to grant or revoke the read privilege. This argument must be specified.
`user`	An optional character string specifying the user that read privilege of the named R script or datastore is granted to or revoked from. The default value `NULL` indicates the privilege is granted to or revoked from public.

Details

Functions `ore.grant` and `ore.revoke` require the user to have the 'RQADMIN' Oracle Database role.

Value

Functions `ore.grant` and `ore.revoke` return an invisible `NULL` value if they succeed in privilege grant or revoke; otherwise they produce an error.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.scriptCreate, ore.scriptDrop, ore.scriptLoad, ore.scriptList, ore.load, ore.datastore, ore.datastoreSummary

Examples

```
if (!interactive())
{
  library(OREds)
  library(OREembed)
  # create an R script for the current user
  ore.scriptCreate("MYLM",
                   function(data, formula, ...) lm(formula, data, ...))
  IRIS <- ore.push(iris)
  ore.tableApply(IRIS[1:4], FUN.NAME = "MYLM",
                 formula = Sepal.Length ~ .)

  # create a global R script available to any user
  ore.scriptCreate("GLBGLM",
                   function(data, formula, ...)
                     glm(formula=formula, data=data, ...),
                   global = TRUE)
  ore.tableApply(IRIS[1:4], FUN.NAME = "GLBGLM",
                 formula = Sepal.Length ~ .)

  # list R scripts
  ore.scriptList()
  ore.scriptList(pattern="LM", type="all")

  # load an R script to an R function object
  ore.scriptLoad(name="MYLM")
  ore.scriptLoad(name="GLBGLM", newname="MYGLM")
  MYLM(iris, formula = Sepal.Length ~ .)
  MYGLM(iris, formula = Sepal.Length ~ .)

  # grant and revoke R script read privilege to and from public
  ore.grant(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")
  ore.revoke(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")

  # drop an R script
  ore.scriptDrop("MYLM")
  ore.scriptDrop("GLBGLM", global=TRUE)

  ore.scriptList(type="all")

  # create grantable datastores
```

```
  ore.save(iris, name="ds_1", grantable=TRUE)
  ore.save(mtcars, name="ds_2", grantable=TRUE)

  # grant the read privilege of one datastore to every user
  ore.grant(name="ds_1", type="datastore", user=NULL)

  # show all the datastores
  ore.datastore(type="all")[,-5L]

  # show the grantable datstores
  ore.datastore(type="grantable")[, -4L]

  # show the datastoe the read privilege for which was granted to other users
  ore.datastore(type="grant")

  # revoke the granted privilege
  ore.revoke(name="ds_1", type="datastore", user=NULL)

  ore.delete(name="ds_1")
  ore.delete(name="ds_2")
}
```

`ore.hash` *Oracle R Enterprise ore.vector Hash Function*

Description

Hashes the values in an `ore.vector` object.

Usage

```
ore.hash(x, ...)
## S4 method for signature 'ore.vector'
ore.hash(x, size = 4294967296,
         seed = round(runif(1, min = -0.5, max = 4294967295.5)), ...)
```

Arguments

`x`	An `ore.vector` object.
`size`	A numeric value within `[1,4294967296]` representing the total number of hash buckets.
`seed`	A numeric value within `[0,4294967295]` representing the seed value for the hash function.
`...`	For future expansion.

Details

For the Oracle Database, the SQL snippet `ORA_HASH(<SQL expr>,size -1,seed) + 1` is used to calculate the hash value.

Value

Returns an `ore.numeric` object containing the hash values for the original vector.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.vector, ore.recode

Examples

```
vec <- ore.push(c("a", "b", "a", NA, "c", "e"))
ore.hash(vec, seed = 123)
```

---

ore.hiveOptions *Set HIVE Options*

Description

Sets HIVE options, namely, field delimiters for the HIVE tables and the current database name.

Usage

```
ore.hiveOptions(delim = "\001", dbname, storedAs='textfile',
                exeEngine = 'mr')
```

Arguments

delim	A character string specifying the field delimiter for the HIVE tables created by ORCH-HIVE.
dbname	A character string specifying the HIVE database name for the current session.
storedAs	A character string specifying the HIVE table storage type for the current session.
exeEngine	A character string specifying the HIVE execution engine for the current session.

Details

Use the delim argument to configure the separator character between fields (columns) in the text files corresponding to the HIVE tables. Use the dbname argument to set the database name for ORCH-HIVE to work on. The database must pre-exist otherwise an error is thrown. This is an environment-level change and once the arguments are set, they are respected by all the tables created using ORCH-HIVE in the R session.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

ore.showHiveOptions

Examples

```
if(ore.is.connected(type="HIVE"))
{
if (!interactive())
  ore.hiveOptions(delim = ",")
}
```

`ore.impalaOptions` *Set IMPALA Options*

Description

Sets IMPALA options, namely, field delimiters for the IMPALA tables and the current database name.

Usage

```
ore.impalaOptions(delim = "\001", dbname, storedAs='textfile',
                  exeEngine = 'mr')
```

Arguments

`delim`	A character string specifying the field delimiter for the IMPALA tables created by ORCH-HIVE.
`dbname`	A character string specifying the IMPALA database name for the current session.
`storedAs`	A character string specifying the IMPALA table storage type for the current session.
`exeEngine`	A character string specifying the IMPALA execution engine for the current session.

Details

Use the `delim` argument to configure the separator character between fields (columns) in the text files corresponding to the IMPALA tables. Use the `dbname` argument to set the database name for ORCH-HIVE to work on. The database must pre-exist otherwise an error is thrown. This is an environment-level change and once the arguments are set, they are respected by all the tables created using ORCH-HIVE in the R session.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

`ore.showImpalaOptions`

Examples

```
if(ore.is.connected(type="IMPALA"))
{
if (!interactive())
  ore.ImpalaOptions(delim = ",")
}
```

---

ore.itemsets-class *Class* ore.itemsets

Description

The ore.itemsets class represents a set of itemsets of an ore.odmAssocRules object in an Oracle database. An ore.odmAssocRules object represents an Association model created by Oracle Data Mining.

Subsetting

In the code snippets below, argument x is an ore.itemsets object.

subset(x,min.support=NUll,max.itemset.length=NULL,min.itemset.length=NULL,items
: Returns a new ore.itemsets object using:

min.support The minimum support of the itemsets to return.

max.itemset.length The maximum length of the itemsets to return.

min.itemsets.length The minimum length of the itemsets to return.

itemset.id The itemset ID (numeric type) of the itemsets to return.

items A list of items. This method returns all itemsets that contain one or more of the specified items. Examples of items lists are items=list("apple","orange") and items =list(age=31,occupation="engineer").

Utilities

In the code snippets below, argument x is an ore.itemsets object.

ore.pull(x): Returns the in-memory counterpart of the ore.itemsets object, which is an itemsets object in the **arules** package.

Details

The ore.itemsets class represents a set of itemsets in an Oracle Database. The set of itemsets is the result of an Association Rules model.

The column names of an ore.itemsets object are "ITEMSET_ID", "NUMBER_OF_ITEMS", "ITEMS", and "SUPPORT".

You can pull itemsets into memory in a local R session from an Oracle Database instance by using ore.pull. The in-memory object is of class itemsets, which is defined in the **arules** package.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.rules, ore.odmAssocRules

Examples

```
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
  item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
  max.rule.length = 3)

itemsets <- itemsets(ar.mod)

sub.itemsets1 <- subset(itemsets, items=list("b", "c"))
sub.itemsets2 <- subset(sub.itemsets1, min.itemset.length=2)

library(arules, warn.conflicts=FALSE)
items.arules <- ore.pull(itemsets)
inspect(items.arules[1:2])
items.arules1 <- ore.pull(sub.itemsets1)
items.arules2 <- ore.pull(sub.itemsets2)
```

---

ore.lazyLoad *Oracle R Enterprise Datastore Lazy Loading Function*

Description

Lazy (just in time) loads R objects from the named Oracle R Enterprise datastore in the user's Oracle Database schema.

Usage

```
ore.lazyLoad(name, list = character(0), envir = parent.frame())
```

Arguments

name	A character string specifying the name of the datastore.
list	An optional character vector containing the names of the objects within the datastore to lazy load. If this argument is not specified, then all of the objects within the datastore will be lazy loaded.
envir	The environment into which the objects are to be lazy loaded.

Details

Unlike the ore.load function, the ore.lazyLoad function does not immediately retrieve the specified objects from an Oracle R Enterprise datastore. Instead objects are retrieved upon first reference to the object. See examples below for implications of lazy loading.

Value

Function ore.lazyLoad returns a character vector specifying the names of objects that are lazy loaded from the datastore.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.save, ore.delete, ore.datastore, ore.datastoreSummary, ore.load

Examples

```
# save object to new datastore
x <- 1:10
ore.save(x, name = "rq$ds_1")

# enable lazy loading of object
e <- new.env()
ore.lazyLoad(name = "rq$ds_1", envir = e)
bindingIsActive("x", e)

# load object
e$x
bindingIsActive("x", e)

# re-enable lazy loading
ore.lazyLoad(name = "rq$ds_1", envir = e)
bindingIsActive("x", e)

# overwrite object in datastore
x <- letters
ore.save(x, name = "rq$ds_1", overwrite = TRUE)

# load updated object
e$x
bindingIsActive("x", e)

# clean up
rm(x, e)
ore.delete(name = "rq$ds_1")
```

ore.list-class *Class* ore.list

Description

An ore class for representing and manipulating list data stored in an Oracle database.

List Methods

x[[i]], x$name: Returns the selected element as an ore.object object.

length(x): Returns a integer value representing the number of elements.

names(x): Returns a character vector containing the element names.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.object

Examples

```
x <- list(A = 1, B = "value", C = TRUE)
X <- ore.push(x)
class(X)          # ore.list
X$A
X[[2]]
X[["C"]]
length(X)
names(X)
X

y <- ore.pull(X)
class(y)          # list
```

`ore.load` *Oracle R Enterprise Datastore Loading Function*

Description

Loads R objects from a datastore in the user's Oracle Database schema.

Usage

```
ore.load(name, list = character(0), owner, envir = parent.frame())
```

Arguments

`name`	A character string specifying the name of the datastore.
`list`	An optional character vector containing the names of the R objects within the datastore to load. If this argument is not specified, then all of the R objects within the datastore will be loaded.
`owner`	An optional character string specifying the owner of the datastore to load. If the owner of the datastore is not the current user, the read privilege for the datastore must be granted to the current user by `ore.grant`; otherwise `ore.load` errors out. If the `owner` argument not specified, the default value is the current user.
`envir`	The environment into which the R objects are to be loaded.

Details

Functions `ore.save` and `ore.load` operate together in a similar manner to functions `save` and `load` from the **base** package. Whereas the `save` and `load` functions save and load R objects to and from an external file, the `ore.save` and `ore.load` functions save and load R objects from a datastore.

Value

Function `ore.load` returns a character vector specifying the names of the R objects that are loaded from the datastore to environment `envir`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.save`, `ore.move`, `ore.delete`, `ore.datastore`, `ore.datastoreSummary`, `ore.lazyLoad`
`ore.grant` `ore.revoke`

Examples

```
if (any(sapply(c("x", "y", "z", "e"), exists)))
  stop("object x, y, z, or e exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

# save all objects in the current workspace environment to
# a datastore with name 'rq$ds_1' in the user's schema
ore.save(list = ls(), name = "rq$ds_1")
rm(x, y, z)

# load all objects from datastore rq$ds_1 to current workspace
ore.load(name = "rq$ds_1")
ls()

# load x and y from datastore rq$ds_1 to environment e
e <- new.env()
ore.load(name = "rq$ds_1", list = c("x", "y"), envir = e)
ls(envir = e)

# clean up
ore.delete(name = "rq$ds_1")
rm(x, y, z, e)
```

---

ore.logical-class *Class* ore.logical

Description

The `ore.logical` class represents `logical` data columns in Oracle R Enterprise.

Group Generics

`ore.logical` objects have support for S4 group generic functionality:

`Logic "&","|"`

See S4groupGeneric for more details.

Logical Methods

`!x`: Returns an `ore.logical` object containing the logical negation (NOT) of argument `x`.

`ifelse(x,yes,no)`: For each element of argument `x`, returns the corresponding element in argument `yes` if `TRUE`, otherwise returns the element in argument `no`. Arguments `yes` and `no` may be length one vectors (scalar values) or `ore.vector` objects.

`xor(x,y)`: Returns an `ore.logical` object containing the exclusive or combination of arguments `x` and `y`.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.matrix, ore.number, ore.vector

Examples

```
x <- c(TRUE, FALSE, TRUE, FALSE, TRUE)
X <- ore.push(x)
class(X)
X

y <- ore.pull(X)
class(y)
```

---

ore.ls *Oracle R Enterprise Object Listing Function*

Description

Returns a vector of character strings giving the names of the ore.frame objects, representing database tables and views, available in the R environment for a schema in the Oracle R Enterprise session.

Usage

```
ore.ls(schema, all.names = FALSE, pattern)
```

Arguments

schema	A character string specifying the database schema name.
all.names	A logical value indicating whether to include objects with a leading . character.
pattern	An optional regular expression whereby only matching names are returned.

Details

If argument schema is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

Value

A character vector.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.attach, ore.connect, ore.exists, ore.get, ore.rm, ore.sync

Examples

```
if (!interactive())
{
  ore.ls()
  ore.ls("RQUSER")
  ore.ls("RQUSER", all.names = TRUE)
  ore.ls("RQUSER", pattern = "IR")
}
```

---

ore.make.names *Oracle R Enterprise Valid Column Name Generator*

Description

Creates valid column names for ore.frame objects.

Usage

```
ore.make.names(names)
```

Arguments

names	A character vector specifying unvalidated names.

Details

Creates distinct strings from argument names that are at most 30 bytes long using an iterative algorithm involving the call

make.unique(abbreviate(names,minlength,strict = TRUE))

where minlength <= 30.

Value

A character vector with the same length as argument names containing valid column names for an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

make.names, abbreviate

Examples

```
xnames <- c("col1", "Col.2", "COL_3", "col 4", "col1",
            "L_A_S_T C.O.L.U.M.N abcdefghijklmnopqrstuvwxyz")
ore.make.names(xnames)
```

---

ore.matrix-class *Class* ore.matrix

Description

The ore.matrix class represents numeric matrices in Oracle R Enterprise.

Accessors

In the code snippets below, argument x is an ore.frame object.

nrow(x), ncol(x): Returns the number of rows and columns, respectively.

NROW(x), NCOL(x): Same as nrow(x) and ncol(x), respectively.

dim(x): Returns an integer vector defined as c(nrow(x),ncol(x)).

rownames(x), colnames(x): Returns the row and column names, respectively.

rownames(x) <-value, colnames(x) <-value: Replaces the row and column names, respectively.

dimnames(x): Returns a list object defined as list(rownames(x),colnames(x)).

dimnames(x) <-value: Replaces the row and column names.

Subsetting

In the code snippets below, argument x is an ore.matrix object.

x[i,j,drop=TRUE]: Returns a new ore.matrix object made of the selected rows and columns. For single column selection, the drop argument specifies whether or not to return an ore.vector object.

Group Generics

ore.matrix objects have support for S4 group generic functionality:

Arith "+", "-", "*", "^", "%%", "%/%", "/"

Math "abs", "sign", "sqrt", "ceiling", "floor", "trunc", "cummax", "cummin", "cumprod", "cumsum", "log", "log10", "log2", "log1p", "acos", "acosh", "asin", "asinh", "atan", "atanh", "exp", "expm1", "cos", "cosh", "sin", "sinh", "tan", "tanh", "gamma", "lgamma", "digamma", "trigamma"

Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.

Matrix Operations

`t(x)`: Returns the matrix transpose of argument `x`.

`crossprod(x,y = NULL)`, `tcrossprod(x,y = NULL)`: Returns the matrix crossproduct of the `x` and `y` arguments.

`x %*% y`: Returns the matrix product of the `x` and `y` arguments.

`colMeans(x,na.rm = FALSE)`: Returns an `ore.number` object containing the column means of argument `x`.

`colSums(x,na.rm = FALSE)`: Returns an `ore.number` object containing the column sums of argument `x`.

`rowMeans(x,na.rm = FALSE)`: Returns an `ore.number` object containing the row means of argument `x`.

`rowSums(x,na.rm = FALSE)`: Returns an `ore.number` object containing the row sums of argument `x`.

`scale(x,center = TRUE,scale = TRUE)`: Returns an `ore.matrix` object containing the possibly centered and scaled version of argument `x`.

`max.col(m,ties.method = c("first","last"))`: Returns an `ore.integer` object containing the column number of the maximum value for each row of argument `m`.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore`, `ore.character`, `ore.datetime`, `ore.factor`, `ore.frame`, `ore.logical`, `ore.number`, `ore.vector`

Examples

```
IRIS_MATRIX <- ore.push(as.matrix(iris[1:4]))
crossprod(IRIS_MATRIX)
```

ore.move *Oracle R Enterprise Datastore Moving Function*

Description

Renames a datastore in the user's Oracle Database schema, moves or renames R objects within a datastore, or both moves and renames R objects within a datastore.

Usage

```
ore.move(name = stop("parameter 'name' must be specified"),
         newname = NULL,
         object.names = NULL,
         object.newnames = NULL,
         overwrite = FALSE)
```

Arguments

name	A character string specifying the datastore to be renamed or modified.
newname	A character string specifying the newname of the datastore or the name of the datastore that is the destination for objects moved from the source datastore. Can be left unspecified if objects are only being renamed and not moved out of the datastore specified by name. Cannot be an empty string.
object.names	A character vector specifying the R objects to be moved or renamed, or both.
object.newnames	A character vector specifying the newnames of the R objects specified by object.names. If not NULL, must be the same length as object.names. Cannot contain an empty string.
overwrite	If TRUE and renaming a datastore, then if the datastore specified by newname already exists, the datastore specified by newname and its contents are overwritten by the contents of the datastore specified by name. If TRUE and moving or renaming datastore R objects, or both, then if the destination datastore already has objects whose names are in object.newnames, those objects are overwritten. If FALSE, then an attempt to rename a datastore or object to an existing datastore or object, or an attempt to move an R object to a datastore where an object of the same name already exists will cause an error. Default value is FALSE.

Details

This function performs one of three actions depending on the input. If object.names or object.newnames is a character vector of length zero, the function treats it as NULL.

1) Rename a datastore in the user's Oracle Database schema. Neither name nor newname may be NULL, and newname may not be an empty string. old.obj.name and object.newnames must both be NULL. The datastore specified by name will be renamed to the value specified by newname.

2) Rename objects within a datastore. name must not be NULL. newname must be either NULL or the same as name. object.names and object.newnames must have distinct values, and neither can be NULL.

3) Move objects from one datastore to another. `name` and `newname` must have distinct values, and neither can be `NULL`. `object.names` cannot be `NULL`. If `object.newnames` is not `NULL` and if the names in it are distinct from `object.names`, the objects are renamed in the datastore specified by `newname`.

An error occurs if there is not a one-to-one mapping of names in `object.names` to `object.newnames`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.save`, `ore.load`, `ore.delete`, `ore.datastore` `ore.datastoreSummary`

Examples

```
if (any(sapply(c("a1", "a2", "b1", "b2", "w1", "w2", "x1", "x2", "z1", "z2"), exists)))
  stop("object a1, a2, b1, b2, w1, w2, x1, x2, z1, or z2 exists")

if (nrow(ore.datastore(pattern="^rq\$ds_[123]$")) > 0)
  stop("datastore 'rq$ds_1', 'rq$ds_2', or 'rq$ds_3' exists")

a1 <- data.frame(a=c("foo", "bar"), b=c("xx", "yy"))
a2 <- data.frame(a=1:10, b=11:20)
b1 <- ore.push(a1)
b2 <- ore.push(a2)
w1 <- stats::runif(20)
w2 <- stats::runif(50)
x1 <- list(a = 1, b = TRUE, c = "oops")
x2 <- list(a = w1, b = w2, c = "data", d=200)
z1 <- ore.push(x1)
z2 <- ore.push(w1)

ore.save(a1, a2, w1, w2, x1, x2, name="rq$ds_1")
ore.save(b1, b2, name="rq$ds_2")
ore.save(z1, z2, name="rq$ds_3")

#rename a datastore
ore.move(name="rq$ds_1", newname="rq$ds_1_renamed")
ds <- ore.datastore()
## Not run:
ds

## End(Not run)


#rename objects within a datastore
ore.move(name="rq$ds_1_renamed", object.names=c("x1", "x2"),
         object.newnames=c("y1", "y2"))
ore.datastoreSummary(name="rq$ds_1_renamed")

#move objects from one datastore to another
```

```
ore.move(name="rq$ds_1_renamed", newname="rq$ds_2", object.names=c("y1", "y2"))
ore.datastoreSummary(name="rq$ds_1_renamed")
ore.datastoreSummary(name="rq$ds_2")

#move objects from one datastore to another, and renaming them
ore.move(name="rq$ds_2", newname="rq$ds_1_renamed", object.names=c("y1", "y2"),
         object.newnames=c("z1", "z2"))
ore.datastoreSummary(name="rq$ds_1_renamed")
ore.datastoreSummary(name="rq$ds_2")

#rename a datastore, overwriting the existing datastore
ore.move(name="rq$ds_1_renamed", newname="rq$ds_2", overwrite = TRUE)
ore.datastoreSummary(name="rq$ds_2")

#rename objects within a datastore, overwriting existing objects of the same name
ore.move(name="rq$ds_2", object.names=c("w1", "w2"), object.newnames=c("z1", "z2"),
         overwrite = TRUE)
ore.datastoreSummary(name="rq$ds_2")

#transfering objects to another datastore, overwriting existing objects of the same name
ore.move(name="rq$ds_2", newname="rq$ds_3", object.names=c("z1", "z2"),
         overwrite=TRUE)
ore.datastoreSummary(name="rq$ds_2")
ore.datastoreSummary(name="rq$ds_3")

#rename objects while transfering them to another datastore, overwriting existing objects
ore.move(name="rq$ds_3", newname="rq$ds_2", object.names=c("z1", "z2"),
         object.newnames=c("a1", "a2"), overwrite=TRUE)
ore.datastoreSummary(name="rq$ds_2")
ore.datastoreSummary(name="rq$ds_3")

ore.delete(name="rq$ds_2")
ore.delete(name="rq$ds_3")
rm(a1, a2, b1, b2, w1, w2, x1, x2, z1, z2)
```

---

ore.number-class *Class* ore.number

Description

The ore.number subclasses ore.integer and ore.numeric represent integer and numeric data columns in Oracle R Enterprise.

Group Generics

ore.number objects have support for S4 group generic functionality:

Arith "+", "-", "*", "^", "%%", "%/%", "/"

Compare "==", ">", "<", "!=", "<=", ">="

Logic "&", "|"

Ops "Arith", "Compare", "Logic"

Math "abs", "sign", "sqrt", "ceiling", "floor", "trunc", "cummax", "cummin", "cumprod", "cumsum", "log", "log10", "log2", "log1p", "acos", "acosh", "asin", "asinh", "atan", "atanh", "exp", "expm1", "cos", "cosh", "sin", "sinh", "tan", "tanh", "gamma", "lgamma", "digamma", "trigamma"

Math2 "round", "signif"

Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.

Numerical Data Methods

cut(x,breaks,labels = NULL,include.lowest = FALSE,right = TRUE,dig.lab = 3L,...): Returns an ore.vector object containing a categorical representation of x-based interval ranges. When labels is FALSE, the output is an ore.integer object; otherwise returns an ore.factor object.

- breaks Either a numeric vector of two or more distinct cut points or an integer value greater than or equal to 2 that specifies the number of intervals to create.
- labels Either a character vector containing the labels for the resulting categories, the logical value FALSE to specify an integer coding for the intervals, or NULL to construct labels using (a,b] or [a,b) interval notation.
- include.lowest A logical value indicating if the lowest (when argument right is TRUE) or highest (when argument right is FALSE) interval should be closed, i.e. of the form [a,b].
- right A logical value indicating if the half-open intervals should be closed on the right or left.
- dig.lab When argument labels is NULL, the number of digits to use in the formatting of the break numbers in the labels.

diff(x,lag = 1,differences = 1): Returns suitably lagged and iterated differences of argument x.

- lag An integer indicating which lag to use.
- differences An integer indicating the order of the difference.

is.finite(x): Returns an ore.logical object indicating which values in argument x contain finite numbers.

is.infinite(x): Returns an ore.logical object indicating which values in argument x contain infinite numbers.

is.nan(x): Returns an ore.logical object indicating which values in argument x contain not-a-number values.

scale(x,center = TRUE,scale = TRUE): Returns an ore.number object containing the possibly centered and scaled version of argument x.

tabulate(bin,nbins = max(1,bin,na.rm = TRUE)): Returns an integer vector containing counts of the number of times each integer occurs in argument bin.

zapsmall(x,digits = getOption("digits")): Returns an ore.number object where values close to 0 are replaced with 0.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.vector

Examples

```
v <- c(2, 4, 4, 7, 7)
V <- ore.push(v)
V
log(V)
tabulate(V)
```

---

ore.number-multivar

Oracle R Enterprise Multivariate Statistics

Description

Performs multivariate numerical aggregation methods for ore.frame objects based on function in R's **stats** package.

Usage

```
  ## S4 method for signature 'ore.frame'
cor(x, y = NULL, use = "everything",
      method = c("pearson", "kendall", "spearman"))
  ## S4 method for signature 'ore.frame'
cov(x, y = NULL, use = "everything",
      method = c("pearson", "kendall", "spearman"))
```

Arguments

x,y	Either argument x is an ore.frame object containing numeric columns and argument y is NULL or arguments x and y are ore.number objects.
use	A method of computation when missing values are present. One of "everything", "all.obs", "complete.obs", or "na.or.complete".
method	For cor method where x and y are ore.number objects, can be one of "pearson", "kendall", "spearman"; otherwise must be "pearson".

Details

These statistics are calculated using a custom analytic based on the crossproduct of the centered (and in the case of `cor` scaled) data matrix.

Unlike the `cor` and `cov` functions in the **stats** package, `use = "pairwise.complete.obs"` and `method %in% c("kendall","spearman")` are not supported.

Value

A matrix of dimension `ncol(x)` by `ncol(x)`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`cor`, `cov`, ore.number-univar, ore.vector-aggregate

Examples

```
LONGLEY <- ore.push(longley)
all.equal(cor(LONGLEY), cor(longley))
all.equal(cov(LONGLEY), cov(longley))
```

---

ore.number-univar *Oracle R Enterprise Univariate Statistics*

Description

Performs univariate numerical aggregation methods for `ore.number` vectors based on the corresponding function in R's **stats** package.

Usage

```
  ## S4 method for signature 'ore.number'
fivenum(x, na.rm = TRUE)
  ## S4 method for signature 'ore.number'
IQR(x, na.rm = FALSE, type = 7)
  ## S4 method for signature 'ore.number'
mad(x, center = median(x), constant = 1.4826,
      na.rm = FALSE, low = FALSE, high = FALSE)
  ## S4 method for signature 'ore.vector'
median(x, na.rm = FALSE)
  ## S4 method for signature 'ore.vector'
quantile(x, probs = seq(0, 1, 0.25),
      na.rm = FALSE, names = TRUE, type = 7, ...)
  ## S4 method for signature 'ore.number'
```

```
sd(x, na.rm = FALSE)
  ## S4 method for signature 'ore.number'
var(x, y = NULL, na.rm = FALSE, use)
```

Arguments

x	An ore.vector object for median and quantile; an ore.number object otherwise.
na.rm	A logical value indicating whether NA values should be ignored during the calculations.
type	The type of quantile algorithm to use according to Types section of quantile. Only 1 and 7 are supported.
center	The center value in median absolute deviation.
constant	The scale factor in median absolute deviation.
low	A logical value indicating whether a 'lo-median' should be calculated in median absolute deviation.
high	A logical value indicating whether a 'hi-median' should be calculated in median absolute deviation.
probs	A numeric vector of probabilities with values in $[0, 1]$.
names	A logical value indicating whether vector element names should be assigned to the quantile values.
y, use	Argument not supported.
...	Additional arguments.

Details

Each of these aggregations are performed within a SQL engine of an Oracle RDBMS. The Oracle SQL aggregate functions involved in these calculations include: AVG, MAX, MEDIAN, MIN, PERCENTILE_CONT, PERCENTILE_DISC, STDDEV_SAMP, and VAR_SAMP.

The fivenum method for ore.number uses combinations of results from Oracle SQL PERCENTILE_DISC to compute Tukey's five number summary of minimum, lower-hinge, median, upper-hinge, and maximum.

For IQR and quantile methods, the type argument serves as a toggle between Oracle SQL PERCENTILE_CONT when type = 7 and Oracle SQL PERCENTILE_DISC when type = 1.

For mad method when the center argument is missing, a two-pass algorithm is used where the first pass calculates the median, represented by center, and the second pass produces constant * median(abs(x -center)).

For sd and var a two-pass algorithm is used where first the pass calculates the mean and the second pass produces STDDEV_SAMP(x -mean) and VAR_SAMP(x -mean).

Value

For IQR, mad, median, sd, and var, returns a numeric vector of length 1.

For fivenum, returns a numeric vector of length 5.

For quantile, returns a numeric vector containing the same number of values as elements in argument probs.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

fivenum, IQR, mad, median, quantile, sd, var, ore.number-multivar, ore.vector-aggregate

Examples

```
# Sample data
X    <- data.frame(V1 = rnorm(100))
oreX <- ore.push(X)

# Test for equality with in-memory data
all.equal(fivenum(oreX$V1), fivenum(X$V1))
all.equal(IQR(oreX$V1), IQR(X$V1))
all.equal(mad(oreX$V1), mad(X$V1))
all.equal(median(oreX$V1), median(X$V1))
all.equal(quantile(oreX$V1), quantile(X$V1))
all.equal(quantile(oreX$V1, type = 1), quantile(X$V1, type = 1))
all.equal(sd(oreX$V1), sd(X$V1))
all.equal(var(oreX$V1), var(X$V1))
```

---

ore.object-class "ore.object" *Objects*

Description

Representation of serialized R objects in an Oracle database.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.list

Examples

```
showClass("ore.object")
```

ore.odmAI *In-Database Attribute Importance Ranking Models*

Description

Computes the relative importance of predictor variables for predicting a response variable using Oracle Data Mining.

Usage

```
ore.odmAI (formula,
           data,
           auto.data.prep = TRUE,
           na.action = na.pass,
           odm.settings = NULL)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ore.frame object used for computing relative importance.
auto.data.prep	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmKMeans.

Details

Attribute Importance uses a Minimum Description Length (MDL) based algorithm that ranks the relative importance of predictor variables in predicting a specified response (target) variable. Use this function to gain insight into the relevance of variables to guide manual variable selection or reduction, with the goal to reduce predictive model build time and/or improve model accuracy.

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms,i.e., column names, to include in the analysis. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. Functions can be applied to response and terms to realize transformations. To exclude columns, use – before each column name to exclude.

Value

An object of class ore.odmAI including the following components:

importance	A data.frame of terms (predictor variables) as row names with their corresponding importance value and rank.
formula	The formula provided when computing the relative importance.
call	The matched call.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVM, ore.odmGLM, ore.odmNB, ore.odmDT, partitions

Examples

```
IRIS <- ore.push(iris)
ore.odmAI(Species ~ ., IRIS)
```

---

ore.odmAssocRules *In-Database Association Rules Models*

Description

Creates an association model using Oracle Data Mining.

Usage

```
ore.odmAssocRules(formula,
       data,
       case.id.column,
       item.id.column = NULL,
       item.value.column = NULL,
       min.support = 0.1,
       min.confidence = 0.1,
       max.rule.length = 4,
       na.action = na.pass,
       odm.settings = NULL)

## S3 method for class 'ore.odmAssocRules'
itemsets(object, ...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of the model specification are given under 'Details'.
data	An ore.frame object used for model building.
case.id.column	The name of a column in data that contains unique case identifiers.

item.id.column
: The name of a column in data that contains item IDs. If NULL, the model treats data as a single-record case relational table, where each row is considered a transaction and column values of that row are converted to items for that transaction; if specified, the model treats data as a transactional or multi-record case table where each row corresponds to an item in the transaction, and the model ignores any columns in data other than the item ID and the item value. The default value is NULL.

item.value.column
: The name of a column in data that contains the value of the item. The default value is NULL.

min.support
: A value that specifies the minimum support for the model.

min.confidence
: A value that specifies the minimum confidence for the model.

max.rule.length
: A value that specifies the maximum rule length for the model.

na.action
: A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings
: Same as odm.settings in ore.odmKMeans.

object
: An object of type ore.odmAssocRules.

...
: Additional arguments.

Details

This function implements the Apriori algorithm (Agrawal and Srikant 1994) to find frequent itemsets and generate association models. It finds the co-occurrence of items in large volumes of "transactional" data such as in the case of market basket analysis, as well as "relational" data. The rule is a statement that the appearance of a set of items in a transactional record implies the existence of another set of items. The groups of items used to form rules must pass a minimum threshold according to how frequently they occur (the support) and how often the consequent follows the antecedent (the confidence). Association models generate all rules that have support and confidence greater than user-specified thresholds. The Apriori algorithm is efficient, and scales well with respect to the number of transactions, number of items, and number of itemsets and rules produced.

The formula specification has the form ~ terms where terms is a series of terms, for example, the column names, to include in the analysis. Multiple terms are specified using + between column names. Use ~ . if all columns in data should be used for model building. To exclude columns, use − before each column name to exclude. Functions can be applied to terms to realize transformations.

Value

The function ore.odmAssocRules returns an object of class "ore.odmAssocRules" that has the following components:

name
: The name of the in-database model.

settings
: A data.frame of settings used to build the model.

attributes
: A named vector of the types of input item values.

inputType
: The type of input data table. It is "trans", "tranWithValue", or "relational" for a multi-record case table, a multi-record case table with the values specified, or a single-record case table, respectively.

formula	A formula specified by users.
call	The matched call.

The method itemsets returns an object of class ore.itemsets that describes the property of each itemset. An ore.itemsets object has the following columns:

ITEMSET_ID	The numerical identifier associated with each itemset.
NUMBER_OF_ITEMS	The number of items in the itemset.
ITEMS	The items in the itemset.
SUPPORT	The support of the itemset.

The method rules returns an object of class ore.rules that describes the property of each rule. For details, see rules.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.rules, ore.itemsets, rules, partitions

Examples

```
# Transactional data.
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

# Build model with specifications.
ar.mod1 <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
  max.rule.length = 3)

# Generate itemsets and rules of the model.
itemsets <- itemsets(ar.mod1)
rules <- rules(ar.mod1)

# subsetting
sub.itemsets <- subset(itemsets, min.support=0.7, items=list("b"))
sub.rules <- subset(rules, min.confidence=0.7, lhs=list("b", "c"))

# Convert the rules to the rules object in arules package.
# The package arules is required.
library(arules, warn.conflicts=FALSE)
rules.arules <- ore.pull(rules)
```

```
inspect(rules.arules[1:3])

# Convert itemsets to the itemsets object in arules package.
itemsets.arules <- ore.pull(itemsets)
inspect(itemsets.arules[1])

if (interactive() && suppressWarnings(require(arulesViz, quietly=TRUE)))
  plot(rules.arules, method = "graph", interactive = TRUE)

# Convert the rules and itemsets objects to data frames.
rules.df <- as(rules.arules, "data.frame")
itemsets.df <- as(itemsets.arules, "data.frame")

# Multi-record case data using item id and item value.
id <- c(1, 1, 2, 2, 3, 3, 3, 4, 4, 5, 5, 6, 6, 7, 8, 8, 9, 9, 10, 11,
  12, 12, 13, 14, 15, 16, 17)
item <- c("a","b","a","b","a","c","d","c","d","c","d","c","d","d","d",
  "a","d","e","a","a","d","e","d","e","d","d","d")
value <- c(1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 2, 3, 2, 1, 2, 3, 1, 2,
  2, 3, 2, 3, 2, 3, 3)
data2.ore <- ore.push(data.frame("ID" = id, "ITEM" = item, "VALUE" = value))

ar.mod2 <- ore.odmAssocRules(~., data2.ore, case.id.column = "ID",
  item.id.column = "ITEM", item.value.column = "VALUE", max.rule.length = 3)

rules <- rules(ar.mod2)
itemsets <- itemsets(ar.mod2)

# Relational data in a single-record case table.
set.seed(7654)
id <- 1:10
color <- sample(c("B", "Y", "W", "G"), 10, replace=TRUE)
shape <- sample(c("tri", "rect", "round"), 10, replace=TRUE)
state <- sample(c("MA", "CA", "NY"), 10, replace=TRUE)
data3.ore <- ore.frame(ID=id, COLOR=color, SHAPE=shape, STATE=state)

ar.mod3 <- ore.odmAssocRules(~., data3.ore, case.id.column = "ID",
  min.support = 0.15, min.confidence = 0.05, max.rule.length = 2)

rules <- subset(rules(ar.mod3), min.confidence=0.5,
  lhs=list(SHAPE="tri", COLOR="B"), orderby="lift")
itemsets <- subset(itemsets(ar.mod3), min.support=0.35)
```

---

ore.odmDT *In-Database Decision Tree Models*

Description

Fits a classification tree using Oracle Data Mining.

Usage

```
ore.odmDT(formula,
       data,
       auto.data.prep = TRUE,
```

```
        cost.matrix = NULL,
        impurity.metric = "gini",
        max.depth = 7,
        min.rec.split = 20,
        min.pct.split = 0.1,
        min.rec.node = 10,
        min.pct.node = 0.05,
        na.action = na.pass,
        odm.settings = NULL)


## S3 method for class 'ore.odmDT'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  bestN = NULL,
  topN.attrs = FALSE,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ore.frame object used for model building.
auto.data.prep	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
cost.matrix	An optional numerical square matrix that specifies the costs for incorrectly predicting the target values. Specifying the row and column names of the matrix is required. The values of the names are possible target values: the row names represent actual target values and the column names represent predicted target values. The vectors of the row and column names must be the same. In general, the diagonal entries of the matrix are zeros. The default value is NULL.
impurity.metric	Tree impurity metric "gini" or "entropy". The default value is "gini".
max.depth	The maximum depth of the tree, from root to leaf inclusive. A value in the range of 2 to 20. The default value is 7.
min.rec.split	The minimum number of cases required in a node for a further split to be possible. The value must be positive. The default value is 20.
min.pct.split	The minimum number of cases required in a node for a further split to be possible. The value is expressed as a percentage of all rows in the training data, and must be in the range of 0 to 20. The default value is 0.1 (0.1 percent).
min.rec.node	The optional minimum number of cases required in a child node. The value must be positive. The default value is 10.
min.pct.node	The optional minimum number of cases required in a child node. The value is expressed as a percentage of the rows in the training data, and must be in the range of 0 to 10. The default value is 0.05 (0.05 percent).

na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmKMeans.
object	An object of type ore.odmDT.
newdata	The data used for scoring.
supplemental.cols	The columns from newdata to include as the columns in the ore.frame prediction result.
type	If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned. The default value is c("class","raw").
bestN	A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.
topN.attrs	boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.
...	Additional arguments affecting the predictions produced.

Details

The Decision Tree algorithm can be used for both binary and multiclass classification problems. The tree structure, created in the model build, is used for a series of simple tests. Each test is based on a single predictor. It is a membership test: either IN or NOT IN a list of values (categorical predictor); or LESS THAN or EQUAL TO some value (numeric predictor).

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. Functions can be applied to response and terms to realize transformations. To exclude columns, use − before each column name to exclude.

The function predict computes predictions based on the input data and model. Results are specified in the section on Value.

Value

The function ore.odmDT returns an object of class ore.odmDT, which includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
distributions	The target class distributions at each tree node.
nodes	The node summary information. See summary.ore.odmDT.
formula	The formula used for the model fitted.

`call` The matched call.

The function `predict` returns an `ore.frame` with columns according to the `type` and `supplemental.cols` parameters. If `type` is `"class"`, the result includes the most likely target class and its probability. If `type` is `"raw"`, the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. If `supplemental.cols` are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmSVM`, `ore.odmGLM`, `ore.odmNB`, `partitions`

Examples

```
m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl  <- as.factor(m$cyl)
m$vs   <- as.factor(m$vs)
m$ID   <- 1:nrow(m)
MTCARS <- ore.push(m)
row.names(MTCARS) <- MTCARS

dt.mod  <- ore.odmDT(gear ~ ., MTCARS)
summary(dt.mod)

dt.res  <- predict (dt.mod, MTCARS,"gear")
with(dt.res, table(gear,PREDICTION))  # generate confusion matrix
```

---

ore.odmEM *In-Database Expectation Maximization Models*

Description

Fits an Expectation Maximization model using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

```
ore.odmEM(formula,
          data,
          num.centers = NULL,
          auto.data.prep = TRUE,
          na.action = na.pass,
          odm.settings = NULL)

## S3 method for class 'ore.odmEM'
histogram(x,
        data=NULL,
        cluster.id="all",...)

## S3 method for class 'ore.odmEM'
predict(object,
        newdata,
        supplemental.cols = NULL,
        type = c("class","raw"),
        na.action = na.pass,
        topN = NULL,
        topN.attrs = FALSE,...)

## S3 method for class 'ore.odmEM'
print(x,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of the model specification are given under 'Details'.
data	An ore.frame object used for model building. This argument is not used in the histogram method.
num.centers	The maximum number of clusters for a clustering model. A value greater than or equal to 1. The default value NULL and the number of clusters is determined by system.
auto.data.prep	A logical value that specifies whether to perform automatic data preparation. The default value is TRUE.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmKMeans.
cluster.id	A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".
object	An object of type ore.odmEM.
x	An object of type ore.odmEM.
newdata	Data used for scoring.
supplemental.cols	The columns to include in the output.

type	If set to "raw", provides the probability for each class returned. If set to "class", the class with the maximum probability is returned.
topN	A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.
topN.attrs	boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.
...	Additional arguments affecting the predictions produced.

Details

The Expectation Maximization algorithm is a popular probability density estimation technique. Oracle Data Mining uses EM to implement a distribution-based clustering algorithm (EM-clustering).

The formula specification has the form ~ terms where terms are the column names to include in the model. Multiple terms are specified using + between column names. Use ~ . if all columns in data should be used for model building. Functions can be applied to terms to realize transformations. To exclude columns, use − before each column name to exclude.

The function histogram generates a grid of histograms with rows corresponding to clusters and columns corresponding to variables, if cluster.id = "all" (default). It provides a quick way to visualize and assess differences among clusters for individual variables. Only leaf clusters (that is, those with no child cluster nodes in the hierarchy) are displayed with cluster.id = "all". If cluster.id is set to the numeric ID of an individual cluster, then only the histograms of variables associated with that cluster are displayed. Both leaf and non-leaf clusters can be displayed using cluster.id.

For specific values associated with the histograms, see clusterhists.ore.odmEM.

Value

The function ore.odmEM produces an object of class "ore.odmEM" that includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), and precision and scale for numeric data.
clusters	A data.frame describing general per-cluster information.
leaf.cluster.count	A data.frame containing leaf clusters with support.
taxonomy	A data.frame describing the parent/child cluster relationship.
centers	A data.frame containing per cluster-attribute center (centroid) information.
centers2	A data.frame containing center information of leaf clusters.
formula	The formula used for the model fitted.
call	The matched call.

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. If type is "class", the result includes the most likely cluster ID and its probability. If type is "raw", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

clusterhists.ore.odmEM, rules.ore.odmEM, ore.odmKMeans, ore.odmOC, partitions

Examples

```
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
         matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push(cbind(data.frame(x), part = as.integer(x[, 2L]*100)%%2L))
em.mod <- ore.odmEM(~. -part, X, num.centers = 3L)
em.mod
summary(em.mod)
rules(em.mod)
clusterhists(em.mod)
histogram(em.mod)

em.res1 <- predict(em.mod,X,type="class",supplemental.cols=c("x","y"))
head(em.res1,3)
em.res1.local <- ore.pull(em.res1)
plot(data.frame(x=em.res1.local$x,
                y=em.res1.local$y),
                col=em.res1.local$CLUSTER_ID)
points(em.mod$centers2, col = rownames(em.mod$centers2), pch = 8, cex=2)

head(predict(em.mod,X))
head(predict(em.mod,X,type=c("class","raw"),supplemental.cols=c("x","y")),3)
head(predict(em.mod,X,type="raw",supplemental.cols=c("x","y")),3)

em.pmod <- ore.odmEM(~. , X, num.centers = 3L,
  odm.settings = list(odms_partition_columns = "part"))
partitions(em.pmod)
summary(em.pmod)
rules(em.pmod)
clusterhists(em.pmod)
histogram(em.pmod, part = "DM$$_P1")
head(predict(em.pmod,X))
```

```
head(predict(em.pmod,X,type=c("class","raw"),supplemental.cols=c("x","y")),3)
head(predict(em.pmod,X,type="raw",supplemental.cols=c("x","y")),3)
```

`ore.odmESA` *In-Database Explicit Semantic Analysis Models*

Description

Creates an Explicit Semantic Analysis model for feature extraction using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

```
ore.odmESA(formula,
        data,
        auto.data.prep = TRUE,
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

## S3 method for class 'ore.odmESA'
features(object,...)

## S3 method for class 'ore.odmESA'
feature_compare(object,
  newdata,
  compare.cols = NULL,
  supplemental.cols = NULL)

## S3 method for class 'ore.odmESA'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  topN = NULL,...)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. For details of the model specification, see 'Details'.
`data`	An `ore.frame` object used for model building.
`auto.data.prep`	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.
`na.action`	A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

odm.settings	Same as odm.settings in ore.odmKMeans. In addition, to perform text mining, parameter CASE_ID_COLUMN_NAME must specify the name of the column containing unique identifiers. Parameters ODMS_TEXT_POLICY_NAME, ODMS_TEXT_MIN_DOCUMENTS, and ODMS_TEXT_MAX_FEATURES can be used to specify the name of a valid Oracle text policy, the minimal number of documents in which echo token occurs, and the maximum number of distinct features for text mining, respectively.
ctx.settings	Same as ctx.settings in ore.odmKMeans.
object	An object of type ore.odmESA.
newdata	Data used for scoring.
compare.cols	Columns used to compare the features. All the columns are used by default.
supplemental.cols	Columns to include in the output.
type	If set to "raw", provides the probability for each feature returned. If set to "class", the feature with the maximum probability is returned.
topN	A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.
...	Additional arguments affecting the predictions produced.

Details

Explicit Semantic Analysis (ESA) is a feature extraction algorithm. ESA does not discover latent features but instead uses explicit features based on an existing knowledge base.

Explicit knowledge often exists in text form. Multiple knowledge bases are available as collections of text documents. These knowledge bases can be generic, for example, Wikipedia, or domain-specific. Data preparation transforms the text into vectors that capture attribute-concept associations.

Value

An object of class "ore.odmESA" includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
formula	The formula used for the model fitted.
call	The matched call.

The function features returns an ore.frame that describes each feature extracted. The columns are:

FEATURE_ID	The numeric identifier associated with the feature.
ATTRIBUTE_NAME	The name of the attribute.
ATTRIBUTE_VALUE	The value of the attribute.
COEFFICIENT	The coefficient associated with the attribute in a particular feature.

The function feature_compare returns an ore.frame containing a SIMILARITY column that measures relatedness and supplementary columns if specified.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVD, ore.odmNMF, partitions

Examples

```
title <- c('Aids in Africa: Planning for a long war',
           'Mars rover maneuvers for rim shot',
           'Mars express confirms presence of water at Mars south pole',
           'NASA announces major Mars rover finding',
           'Drug access, Asia threat in focus at AIDS summit',
           'NASA Mars Odyssey THEMIS image: typical crater',
           'Road blocks for Aids')

# TEXT contents in character column
df <- data.frame(CUST_ID = seq(length(title)), TITLE = title)
ESA_TEXT <- ore.push(df)

# TEXT contains in clob column
attr(df$TITLE, "ora.type") <- "clob"
ESA_TEXT_CLOB <- ore.push(df)

# create text policy (CTXSYS.CTX_DDL privilege is required)
ore.exec("Begin ctx_ddl.create_policy('ESA_TXTPOL'); End;")

# specify TEXT POLICY_NAME, MIN_DOCUMENTS, MAX_FEATURES and
# ESA algorithm settings in odm.settings
esa.mod <- ore.odmESA(~ TITLE, data = ESA_TEXT_CLOB,
  odm.settings = list(case_id_column_name = "CUST_ID",
                      ODMS_TEXT_POLICY_NAME = "ESA_TXTPOL",
                      ODMS_TEXT_MIN_DOCUMENTS = 1,
                      ODMS_TEXT_MAX_FEATURES = 3,
                      ESAS_MIN_ITEMS = 1,
                      ESAS_VALUE_THRESHOLD = 0.0001,
                      ESAS_TOPN_FEATURES = 3))
class(esa.mod)
summary(esa.mod)
settings(esa.mod)
features(esa.mod)
predict(esa.mod, ESA_TEXT, type = "class", supplemental.cols = "TITLE")

# use ctx.settings to specify a character column as TEXT and
# the same above settings as well as TOKEN_TYPE
esa.mod2 <- ore.odmESA(~ TITLE, data = ESA_TEXT,
  odm.settings = list(case_id_column_name = "CUST_ID", ESAS_MIN_ITEMS = 1),
```

```
  ctx.settings = list(TITLE =
    "TEXT(POLICY_NAME:ESA_TXTPOL)(TOKEN_TYPE:STEM)(MIN_DOCUMENTS:1)(MAX_FEATURES:3)"))
summary(esa.mod2)
settings(esa.mod2)
features(esa.mod2)
predict(esa.mod2, ESA_TEXT_CLOB, type = "class", supplemental.cols = "TITLE")

ore.exec("Begin ctx_ddl.drop_policy('ESA_TXTPOL'); End;")
```

`ore.odmESM` *In-Database ESM Models*

Description

Fits a ESM model using Oracle Data Mining,

Usage

```
ore.odmESM(formula,
       data,
       auto.data.prep = TRUE,
       na.action = na.pass,
       odm.settings = NULL,
       ctx.settings = NULL)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
`data`	An `ore.frame` object used for model building.
`auto.data.prep`	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is `TRUE`.
`na.action`	A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.
`odm.settings`	Same as `odm.settings` in `ore.odmESM`.
`ctx.settings`	Text mining settings.

Details

Exponential Smoothing Models(ESM) is a useful technique for extracting meaningful and interpretable features.

The `formula` specification has the form `attribute ~ terms` where `attribute` is the attribute to be used for model build and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use `response ~ .` if all columns in `data` should be used for model building. To exclude columns, use − before each column name to exclude.

Value

The ore.odmESM function returns an object of class ore.odmESM, which includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of variables (attribiutes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
formula	The formula used for the model fitted.
predictions	Predictions based on the input data and model.
call	The matched call.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVM, ore.odmGLM, ore.odmDT, partitions

Examples

```
set.seed(7654)
N <- 10
ts0 <- data.frame(ID=1:N, VAL=runif(N))
ts0.ore <- ore.push(ts0)
mod <- ore.odmESM(VAL~.,ts0.ore, odm.settings = list(case_id_column_name = "ID"))
mod
summary(mod)
```

---

ore.odmGLM *In-Database Generalized Linear Models*

Description

Fits a linear or logistic regression model using Oracle Data Mining. Supports ridge estimation of the coefficients.

Usage

```
ore.odmGLM(formula,
           data,
           weights = NULL,
           type = c("normal", "logistic"),
           na.treatment = c("delete.row", "mean.or.mode"),
           reference = NULL,
           ridge = FALSE,
           ridge.value = NULL,
           ridge.vif = FALSE,
           auto.data.prep = FALSE,
           odm.settings = NULL,
           ctx.settings = NULL)

## S3 method for class 'ore.odmGLM'
residuals(object, type = c("deviance", "pearson", "response"), ...)

## S3 method for class 'ore.odmGLM'
fitted(object, ...)

## S3 method for class 'ore.odmGLM'
predict(object, newdata, supplemental.cols = NULL,
        confint = FALSE, level = 0.95, na.action = na.pass, topN.attrs = FALSE,

## S3 method for class 'ore.odmGLM'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'ore.odmGLM'
deviance(object, ...)

## S3 method for class 'ore.odmGLM'
extractAIC(fit, scale = 0, k = 2, ...)

## S3 method for class 'ore.odmGLM'
logLik(object, ...)

## S3 method for class 'ore.odmGLM'
nobs(object, ...)
```

Arguments

formula	An object of class formula that describes the model to be fit. (See the 'Details' section for more information.)
data	An ore.frame object containing the variables for the model fit.
weights	An optional character string representing the column name in the data argument to use as analytical weights in the model fit.
type	For ore.odmGLM, the type of generalized linear model; either "normal" (Gaussian) or "logistic" (binomial). For the residuals method, the type of residuals; either "deviance", "pearson", or "response".
na.treatment	The missing value treatment; either "delete.row" (delete entire row) or

	"mean.or.mode" (replace missing values with the mean in numeric predictors and the mode in categorical predictors).
reference	An optional response variable category to use as the reference value (non-case/failure code) in a logistic regression model. By default, reference is taken to be the category with the highest prevalence.
ridge	A logical indicator for whether to use ridge estimation for the coefficients.
ridge.value	When the ridge argument is TRUE, an optional positive value for the ridge parameter. If ridge estimation is used and the argument ridge.value is NULL, the ridge parameter is determined by the algorithm.
ridge.vif	(Linear regression only) Optional logical indicator for whether to produce Variance Inflation Factor (VIF) statistics for the ridge estimates. VIFs can only be produced if enough Oracle database system resources are available.
auto.data.prep	A logical indicator for optional automatic data preparation.
object, fit	Object of type ore.odmGLM.
newdata	Data used for scoring.
supplemental.cols	A character vector containing additional columns from argument newdata to include in the prediction result.
confint	A logical indicator for whether to produce confidence intervals for the predicted values.
level	A numeric value within [0, 1] to use for the confidence level.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
topN.attrs	boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.
odm.settings	Same as odm.settings in ore.odmKMeans.
ctx.settings	Same as ctx.settings in ore.odmKMeans.
parm	An optional character vector that specifies which coefficients to include in the set of confidence intervals.
scale	An optional numeric scale parameter.
k	An optional numeric weight of the equivalent degrees of freedom.
...	Additional arguments.

Details

ODM generalized linear models (GLM) implements Gaussian linear regression for continuous response variables and logistic regression for binary response variables using the canonical link and variance functions. The ODM GLM implementation contains two features for robust model fitting: ridge estimation of the coefficients (argument ridge) and automatic data preparation (argument auto.data.prep). When ridge estimation is enabled, a user can supply a specific ridge parameter in argument ridge.value or have an automated procedure determine a suitable value. In the case of exact multi-colinearity, if the user disables ridge estimation, a database exception is raised

to inform the user that the data is problematic and either explanatory variables could be removed from the model or ridge estimation could be used. The automated data preparation includes normalization for numerical values, encoding as a series of indicators for categorical values, and missing data replacement.

In addition to the classical weighted least squares estimation for linear regression and iteratively reweighted least squares estimation for logistic regression, both solved through Cholesky decomposition and matrix inversion, ODM GLM provides a conjugate gradient-based optimization algorithm that does not require matrix inversion and is very well suited to high-dimensional data similar to the approach in [Komarek, 2004]. The choice of algorithm is handled internally and is transparent to the user.

Feature selection and generation can be enabled and configured by the settings in `odm.settings`. The feature selection criteria can be `AIC`, `SBIC`, `RIC`, or `alpha-investing`. The maximum number of features can be specified. Features can be pruned in the final model. Feature generation is only possible when feature selection is enabled. The feature generation method can be either `quadratic` or `cubic`. Note that feature selection and ridge regression are mutually exclusive.

The `formula` argument in function `ore.odmGLM` uses the form `response ~ terms`, where `response` is the numeric or binary category response vector and `terms` is a combination of explanatory variable terms to include in the model. Explanatory terms are separated by the + operator. The `.` terms wildcard is supported with `response ~ .` representing the use of all but the response column in `data` as linear terms in the model. The − operator can be used in combination with `.` to exclude variables from the model. Interaction effects created using the operators `:`, $*$, and $*$ are not supported.

Value

The function `ore.odmGLM` returns an object of class `ore.odmGLM`, which includes the following components:

`coefficients`	A named vector of coefficients.
`residuals`	An `ore.frame` containing three types of residuals: `"deviance"`, `"pearson"`, and `"response"`.
`fitted.values`	An `ore.vector` containing the fitted values.
`rank`	The numeric rank of the fitted model.
`type`	The type of model fit.
`deviance`	Minus twice the maximized log-likelihood, up to a constant.
`aic`	The same version of Akaike's *An Information Criterion* as used by `glm`.
`null.deviance`	The deviance for the null (intercept only) model.
`prior.weights`	The weights initially supplied or 1.
`df.residual`	The residual degrees of freedom.
`df.null`	The residual degrees of freedom for the null model.
`y`	An `ore.vector` containing the response variable.
`converged`	The indicator for whether the model converged.
`model`	An `ore.frame` containing the model frame.
`na.treatment`	An indicator for how missing values were treated.
`na.action`	The number of rows with missing values that were removed.

call	The matched call.
formula	The formula supplied.
terms	The terms object used.
data	The data argument.
nonreference	For logistic regression, the response values that represents success.
ridge	The ridge argument.
auto.data.prep	The auto.data.prep argument.
fit.name	The internal name for the in-database model.
fit.details	The model fit details.

The residuals method returns an ore.vector containing the type of residuals specified in the type argument.

The predict method returns an ore.frame containing the predictions along with the columns specified by the supplemental.cols argument.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Dobson, Annette J. and Barnett, Adrian G. (2008) An Introduction to Generalized Linear Models, Third Edition. Texts in Statistical Science ,77 . Chapman & Hall/CRC Press, Boca Raton, FL.

Komarek, Paul, "Logistic Regression for Data Mining and High-Dimensional Classification" (2004). Robotics Institute. Paper 222.

B. L. Milenova, J. S. Yarmus, and M. M. Campos. SVM in oracle database 10g: removing the barriers to widespread adoption of support vector machines. In Proceedings of the "31st international Conference on Very Large Data Bases" (Trondheim, Norway, August 30 - September 02, 2005). pp1152-1163, ISBN:1-59593-154-6.

Milenova, B.L. Campos, M.M., Mining high-dimensional data for information fusion: a database-centric approach 8th International Conference on Information Fusion, 2005. Publication Date: 25-28 July 2005. ISBN: 0-7803-9286-8. John Shawe-Taylor & Nello Cristianini. Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000.

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

summary.ore.odmGLM, residuals, fitted, predict, confint, deviance, extractAIC, logLik, nobs, ore.odmSVM, ore.odmNB, ore.odmDT, partitions

Examples

```
library(OREstats)
# Linear regression using the longley data set
LONGLEY <- ore.push(longley)
longfit1 <- ore.odmGLM(Employed ~ ., data = LONGLEY)
summ.mod1 <- summary(longfit1)

# Ridge regression using the longley data set
longfit2 <- ore.odmGLM(Employed ~ ., data = LONGLEY, ridge = TRUE,
                       ridge.vif = TRUE)
summ.mod2 <- summary(longfit2)

# Logistic regresion using the infert data set
INFERT <- ore.push(infert)
infit1 <- ore.odmGLM(case ~ age+parity+education+spontaneous+induced,
                     data = INFERT, type = "logistic")
infit1

# Changing the referece value to 1
infit2 <- ore.odmGLM(case ~ age+parity+education+spontaneous+induced,
                     data = INFERT, type = "logistic", reference = 1)
infit2

# Feature selection
IRIS <- ore.push(iris)
irisfit1 <- ore.odmGLM(Sepal.Length ~ ., data = IRIS,
  odm.settings = list(GLMS_FTR_SELECTION = "GLMS_FTR_SELECTION_ENABLE",
                      GLMS_MAX_FEATURES = 5))
settings(irisfit1)
iris.sum1 <- summary(irisfit1)

# Feature extraction
irisfit2 <- ore.odmGLM(Sepal.Length ~ ., data = IRIS,
  odm.settings = list(GLMS_FTR_SELECTION = "GLMS_FTR_SELECTION_ENABLE",
                      GLMS_FTR_GENERATION = "GLMS_FTR_GENERATION_ENABLE",
                      GLMS_MAX_FEATURES = 10))
settings(irisfit2)
iris.sum2 <- summary(irisfit2)
```

---

ore.odmKMeans *In-Database Hierarchical K-Means Models*

Description

Fits a hierarchical k-Means model using Oracle Data Mining.

Usage

```
ore.odmKMeans(formula,
       data,
       auto.data.prep = TRUE,
       num.centers = 10,
       block.growth = 2,
       conv.tolerance = 0.01,
```

```
        distance.function = "euclidean",
        iterations = 3,
        min.pct.attr.support = 0.1,
        num.bins = 10,
        split.criterion = "variance",
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

## S3 method for class 'ore.odmKMeans'
histogram(x,
        data=NULL,
        cluster.id="all",...)

## S3 method for class 'ore.odmKMeans'
predict(object,
        newdata,
        supplemental.cols = NULL,
        type = c("class","raw"),
        na.action = na.pass,
        topN = NULL,
        topN.attrs = FALSE,...)

## S3 method for class 'ore.odmKMeans'
print(x,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of the model specification are given under 'Details'.
data	An ore.frame object used for model building. This argument is not used in the histogram method.
auto.data.prep	A logical value that specifies whether to perform automatic data preparation. The default value is TRUE.
num.centers	The number of clusters for a clustering model. A value greater than or equal to 1. The default value is 10.
block.growth	The numeric growth factor for memory to hold cluster data. The value must be greater than 1 and less than or equal to 5. The default value is 2.
conv.tolerance	The convergence tolerance setting. The value must be greater than 0 and less than or equal to 0.5. The default value is 0.01.
distance.function	A distance function, which can be "cosine", "euclidean", or "fast.cosine". The default value is "euclidean".
iterations	The number of iterations. A value in the range of 1 to 20. The default value is 3.
min.pct.attr.support	The minimum percent required for attributes (variables) to appear in rules. A value in the range 0 to 1. The default value is 0.1.
num.bins	The number of histogram bins. A value greater than 0. The default value is 10.

split.criterion
: A logical value that specifies whether to split clusters by "variance" or "size" for split criteria. The default value is "variance".

na.action
: A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings
: A list to specify Oracle Data Mining parameter settings. This argument is applicable to building a model in Database 12.2 or later. Each list element's name and value refer to the parameter setting name and value, respectively. The setting value must be numeric or string. Refer to Oracle Data Mining User's Guide for each algorithm's valid settings.

  The name of the created Oracle Database mining model is system-determined by default while the parameter MODEL_NAME can be specified to explicitly set the mining model name. Note that ORE does not manage the life cycle of an explicitly-named database mining model.

  When parameter ODMS_PARTITION_COLUMNS is set to the names of the partition columns, then a partition model with sub-model in each partition is created from the input data.

  ore.odmGLM, ore.odmESA, ore.odmKMeans, ore.odmNMF, ore.odmSVD, and ore.odmSVM models can be used to perform text mining. Parameters ODMS_TEXT_POLICY_NAME, ODMS_TEXT_MIN_DOCUMENTS, and ODMS_TEXT_MAX_FEATU can specify the name of a valid Oracle text policy, the minimal number of documents in which echo token occurs, and the maximum number of distinct features for text mining, respectively. Refer to Mining Unstructured Text in Oracle Data Mining User's Guide for details.

ctx.settings
: A list to specify Oracle Text attribute-specific settings. This argument is applicable to building models in Database 12.2 or later. The name of each list element refers to the text column while the list value is a scalar string specifying the attribute-specific text transformation. The valid entries in the string include TEXT, POLICY_NAME, TOKEN_TYPE, and MAX_FEATURES. See Mining Unstructured Text in Oracle Data Mining User's Guide for details.

cluster.id
: A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".

object
: An object of type ore.odmKMeans.

x
: An object of type ore.odmKMeans.

newdata
: Data used for scoring.

supplemental.cols
: The columns to include in the output.

type
: If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.

topN
: A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.

topN.attrs
: boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

...
: Additional arguments affecting the predictions produced.

Details

The k-Means algorithm uses a distance-based similarity measure and tessellates the data space creating hierarchies. It handles large data volumes through summarization and supports sparse data. It is especially useful when the dataset has a moderate number of numerical attributes and there is a predetermined number of clusters.

The formula specification has the form ~ terms where terms are the column names to include in the model. Multiple terms are specified using + between column names. Use ~ . if all columns in data should be used for model building. Functions can be applied to terms to realize transformations. To exclude columns, use − before each column name to exclude.

The function histogram generates a grid of histograms with rows corresponding to clusters and columns corresponding to variables, if cluster.id = "all" (default). It provides a quick way to visualize and assess differences among clusters for individual variables. That is, Only leaf clusters (for example, those with no child cluster nodes in the hierarchy) are displayed with cluster.id = "all". If cluster.id is set to the numeric ID of an individual cluster, then only the histograms of variables associated with that cluster are displayed. Both leaf and non-leaf clusters can be displayed using cluster.id.

For specific values associated with the histograms, see clusterhists.ore.odmKMeans.

Value

The function ore.odmKmeans produces an object of class "ore.odmKMeans" that includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), and precision and scale for numeric data.
clusters	A data.frame describing general per-cluster information.
leaf.cluster.count	A data.frame containing leaf clusters with support.
taxonomy	A data.frame describing the parent/child cluster relationship.
centers	A data.frame containing per cluster-attribute center (centroid) information.
centers2	A data.frame containing center information of leaf clusters.
formula	The formula used for the model fitted.
call	The matched call.

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. If type is "class", the result includes the most likely cluster ID and its probability. If type is "raw", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

clusterhists.ore.odmKMeans, rules.ore.odmKMeans, ore.odmOC, ore.odmEM, partitions

Examples

```
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
         matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push (data.frame(x))
km.mod1 <- NULL
km.mod1 <- ore.odmKMeans(~., X, num.centers=2)
km.mod1
summary(km.mod1)
rules(km.mod1)
clusterhists(km.mod1)
histogram(km.mod1)

km.res1 <- predict(km.mod1,X,type="class",supplemental.cols=c("x","y"))
head(km.res1,3)
km.res1.local <- ore.pull(km.res1)
plot(data.frame(x=km.res1.local$x,
                y=km.res1.local$y),
                col=km.res1.local$CLUSTER_ID)
points(km.mod1$centers2, col = rownames(km.mod1$centers2), pch = 8, cex=2)

head(predict(km.mod1,X))
head(predict(km.mod1,X,type=c("class","raw"),supplemental.cols=c("x","y")),3)
head(predict(km.mod1,X,type="raw",supplemental.cols=c("x","y")),3)

# Text mining with ore.odmKMeans
title <- c('Aids in Africa: Planning for a long war',
           'Mars rover maneuvers for rim shot',
           'Mars express confirms presence of water at Mars south pole',
           'NASA announces major Mars rover finding',
           'Drug access, Asia threat in focus at AIDS summit',
           'NASA Mars Odyssey THEMIS image: typical crater',
           'Road blocks for Aids')
response <- c('Aids', 'Mars', 'Mars', 'Mars', 'Aids', 'Mars', 'Aids')

# TEXT contents in character column
KM_TEXT <- ore.push(data.frame(CUST_ID = seq(length(title)),
                               RESPONSE = response, TITLE = title))

# create text policy (CTXSYS.CTX_DDL privilege is required)
ore.exec("Begin ctx_ddl.create_policy('ESA_TXTPOL'); End;")

# specify POLICY_NAME, MIN_DOCUMENTS, MAX_FEATURES and
```

```
# Text column attribute specification
km.mod <- ore.odmKMeans( ~ TITLE, data = KM_TEXT, num.centers = 2L,
  odm.settings = list(ODMS_TEXT_POLICY_NAME = "ESA_TXTPOL",
                      ODMS_TEXT_MIN_DOCUMENTS = 1,
                      ODMS_TEXT_MAX_FEATURES = 3,
                      kmns_distance="dbms_data_mining.kmns_cosine",
                               kmns_details = "kmns_details_all"),
  ctx.settings = list(TITLE="TEXT(TOKEN_TYPE:STEM)"))
summary(km.mod)
settings(km.mod)
print(predict(km.mod, KM_TEXT, supplemental.cols = "RESPONSE"), digits = 3L)

ore.exec("Begin ctx_ddl.drop_policy('ESA_TXTPOL'); End;")
```

`ore.odmNB` *In-Database Naive Bayes Models*

Description

Fits a Naive Bayes classifier using Oracle Data Mining,

Usage

```
ore.odmNB(formula,
       data,
       auto.data.prep = TRUE,
       class.priors = NULL,
       na.action = na.pass,
       odm.settings = NULL)

## S3 method for class 'ore.odmNB'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  bestN = NULL,
  topN.attrs = FALSE,...)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
`data`	An `ore.frame` object used for model building.
`auto.data.prep`	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is `TRUE`.
`class.priors`	An optional named numerical vector that specifies the priors for the target classes. The names of the vector are the possible target classes. The default value is `NULL`

na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmKMeans.
object	An object of type ore.odmNB.
newdata	The data used for scoring.
supplemental.cols	The columns from newdata to include as columns in the ore.frame prediction result.
type	If set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned. The default value is c("class","raw").
bestN	A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.
topN.attrs	boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.
...	Additional arguments affecting the predictions produced.

Details

The Naive Bayes (NB) algorithm for classification makes predictions using Bayes' Theorem assuming that each attribute is conditionally independent of the others given a particular value of the target (Duda, Hart and Stork 2000). NB provides a flexible general classifier for fast model building and scoring that can be used for both binary and multiclass classification problems.

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. To exclude columns, use − before each column name to exclude.

The function predict computes predictions based on the input data and model. Results are specified in the section on Value.

Value

The ore.odmNB function returns an object of class ore.odmNB, which includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of variables (attribiutes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
apriori	The class distribution for the dependent variable.
tables	A list of tables, one for each predictor variable with conditional probabilities.
levels	A vector of unqiue target class values.
formula	The formula used for the model fitted.

`call` The matched call.

The function `predict` returns an `ore.frame` with columns according to the `type` and `supplemental.cols` parameters. If `type` is `"class"`, the result includes the most likely target class and its probability. If `type` is `"raw"`, the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. If `supplemental.cols` are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmSVM`, `ore.odmGLM`, `ore.odmDT`, `partitions`

Examples

```
# mtcars example
m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl  <- as.factor(m$cyl)
m$vs   <- as.factor(m$vs)
m$ID   <- 1:nrow(m)
MTCARS <- ore.push(m)
row.names(MTCARS) <- MTCARS

nb.mod  <- ore.odmNB(gear ~ ., MTCARS)
summary(nb.mod)

nb.res  <- predict (nb.mod, MTCARS,"gear")
with(nb.res, table(gear,PREDICTION))  # generate confusion matrix

# iris example
IRIS <- ore.push(iris)
species.priors <- c(setosa=0.33, versicolor=0.33, virginica=0.34)

nb.mod  <- ore.odmNB(Species ~ ., IRIS, class.priors=species.priors)
summary(nb.mod)
nb.res  <- predict (nb.mod, IRIS, "Species")
with(nb.res, table(Species, PREDICTION))  # generate confusion matrix
```

`ore.odmNMF` *In-Database Non-Negative Matrix Factorization Models*

Description

Creates a Non-Negative Matrix Factorization model for feature extraction using Oracle Data Mining.

Usage

```
ore.odmNMF(formula,
        data,
        auto.data.prep = TRUE,
        num.features = NULL,
        conv.tolerance = NULL,
        num.iter = NULL,
        rand.seed = NULL,
        allow.negative.scores = FALSE,
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

features(object,...)
## S3 method for class 'ore.odmNMF'
features(object,...)

## S3 method for class 'ore.odmNMF'
feature_compare(object,
  newdata,
  compare.cols = NULL,
  supplemental.cols = NULL)

## S3 method for class 'ore.odmNMF'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  topN = NULL,
  topN.attrs = FALSE,...)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
`data`	An `ore.frame` object used for model building.
`auto.data.prep`	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.

num.features An integer that specifies the number of features to be extracted. The default is estimated by the algorithm.

conv.tolerance
A positive fractional number less than or equal to 0.5 that specifies the convergence tolerance. The default value is 0.05.

num.iter An integer number less than or equal to 500 that specifies the maximum allowed number of iterations. The default value is 50.

rand.seed A number that specifies the random seed. The default value is -1.

allow.negative.scores
A logical value that specifies whether negative scores are allowed in scoring. The default value is FALSE.

na.action A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings Same as odm.settings in ore.odmKMeans.

ctx.settings Same as ctx.settings in ore.odmKMeans.

object An object of type ore.odmNMF.

newdata Data used for scoring.

compare.cols Columns used to compare the features. All the columns are included by default.

supplemental.cols
Columns to include in the output.

type If set to "raw", provides the probability for each feature returned. If set to "class", the feature with the maximum probability is returned.

topN A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.

topN.attrs boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

... Additional arguments affecting the predictions produced.

Details

Non-Negative Matrix Factorization (NMF) is a feature extraction algorithm. Each feature extracted by NMF is a linear combination of the original attribution set. Each feature has a set of non-negative coefficients, which are a measure of the weight of each attribute on the feature. If the argument allow.negative.scores is TRUE, negative coefficients are allowed.

Value

An object of class "ore.odmNMF" includes the following components:

name The name of the in-database model.

settings A data.frame of settings used to build the model.

attributes A data.frame of attributes used to build the model.

formula The formula used for the model fitted.

call The matched call.

The function `features` returns a `data.frame` to describe each feature extracted. The columns are:

`FEATURE_ID` The numeric identifier associated with the feature.

`ATTRIBUTE_NAME`
The name of the attribute.

`ATTRIBUTE_VALUE`
The value of the attribute.

`COEFFICIENT` The coefficient associated with the attribute in a particular feature.

The function `feature_compare` returns an `ore.frame` containing the `SIMILARITY` column and supplementary columns if specified.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmSVD`, `ore.odmESA`, `partitions`

Examples

```
library(OREstats)
training.set <- ore.push(npk[1:18, c("N","P","K")])
scoring.set <- ore.push(npk[19:24, c("N","P","K")])

nmf.mod <- ore.odmNMF(~., training.set, num.features = 3)
features(nmf.mod)
summary(nmf.mod)
predict(nmf.mod, scoring.set)
```

---

ore.odmNN *In-Database Neural Network Models*

Description

Fits a neural network model using Oracle Data Mining,

Usage

```
ore.odmNN(formula,
       data,
       type,
       auto.data.prep = TRUE,
       na.action = na.pass,
       odm.settings = NULL,
       ctx.settings = NULL)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ore.frame object used for model building.
type	For ore.odmNN, the type of NN model: "classification" or "regression".
auto.data.prep	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmNN.
ctx.settings	Text mining settings.

Details

An Neural Network(NN) algorithm is a non-linear predictive algorithm that learn through training. It attempt to emulate the processing of a biological brain. It could be used for classification or regression.

The formula specification has the form attribute ~ terms where attribute is the attribute to be used for model build and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. To exclude columns, use − before each column name to exclude.

Value

The ore.odmNN function returns an object of class ore.odmNN, which includes the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of variables (attribiutes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
formula	The formula used for the model fitted.
nlayers	Number of layers of the fitted model
nnodes	Number of nodes in each layer
weight	Weights of fitted model
call	The matched call.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVM, ore.odmGLM, ore.odmDT, partitions

Examples

```
INFERT <- ore.push(infert)
fit <- ore.odmNN(case~., data=INFERT, type='regression')
fit
summary(fit)
head(weights(fit),3)
ans <- predict(fit, INFERT)
head(ans,3)
fit <- ore.odmNN(education~., data=INFERT, type='classification')
fit
summary(fit)
head(weights(fit),3)
ans <- predict(fit, INFERT)
head(ans,3)
```

---

ore.odmOC *In-Database Orthogonal Partitioning Cluster Models*

Description

Creates an Orthogonal Partitioning Cluster (O-Cluster) model using Oracle Data Mining.

Usage

```
ore.odmOC(formula,
       data,
       num.centers = 10,
       max.buffer = 50000,
       sensitivity = 0.5,
       na.action = na.pass,
       odm.settings = NULL)

## S3 method for class 'ore.odmOC'
histogram(x,
        data=NULL,
        cluster.id="all",...)
```

```
## S3 method for class 'ore.odmOC'
predict(object,
        newdata,
        supplemental.cols = NULL,
        type = c("class","raw"),
        na.action = na.pass,
        topN = NULL,
        topN.attrs = FALSE,...)

## S3 method for class 'ore.odmOC'
print(x,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ordered ore.frame object used for model building. The number of ordering columns must be 1. This argument in the method histogram is unused.
num.centers	A number greater than or equal to 1 that specifies the number of clusters for the clustering model. The default value is 10.
max.buffer	A number greater than or equal to 1 that specifies the maximum buffer size. The default value is 50,000.
sensitivity	A fraction that specifies the peak density required for separating a new cluster. The fraction is related to the global uniform density. The value must be in the range 0 to 1. The default value is 0.5.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmKMeans.
cluster.id	A numerical cluster ID that specifies a particular cluster shown in the histograms, or "all" to show all the clusters. The default value is "all".
object	An object of type ore.odmOC.
x	An object of type ore.odmOC.
newdata	The data used for scoring.
supplemental.cols	The columns to include in the output.
type	If set to "raw", provides the probability for each class returned. If set to "class", the class with the maximum probability is returned.
topN	A positive integer that restricts the set of predicted clusters to those that have one of the specified number of highest probability values.
topN.attrs	boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.
...	Additional arguments affecting the predictions produced.

Details

The O-Cluster algorithm creates a hierarchical grid-based clustering model, that is, it creates axis-parallel (orthogonal) partitions in the input attribute space. The algorithm operates recursively. The resulting hierarchical structure represents an irregular grid that tessellates the attribute space into clusters. The resulting clusters define dense areas in the attribute space.

The clusters are described by intervals along the attribute axes and the corresponding centroids and histograms. A parameter called `sensitivity` defines a baseline density level. Only areas with peak density above this baseline level can be identified as clusters.

The k-Means algorithm tessellates the space even when natural clusters may not exist. For example, if there is a region of uniform density, k-Means tessellates it into n clusters (where n is specified by the user). O-Cluster separates areas of high density by placing cutting planes through areas of low density. O-Cluster needs multi-modal histograms (peaks and valleys). If an area has projections with uniform or monotonically changing density, O-Cluster does not partition it.

The clusters discovered by O-Cluster are used to generate a Bayesian probability model that is then used during scoring (`predict`) for assigning data points to clusters. The generated probability model is a mixture model where the mixture components are represented by a product of independent normal distributions for numerical attributes and multinomial distributions for categorical attributes.

Keep the following in mind if you choose to prepare the data for O-Cluster: 1. O-Cluster does not necessarily use all the input data when it builds a model. It reads the data in batches (the default batch size is 50000). It will only read another batch if it believes, based on statistical tests, that there may still exist clusters that it has not yet uncovered. 2. Because O-Cluster may stop the model build before it reads all of the data, it is highly recommended that the data be randomized. 3. Binary attributes should be declared as categorical. O-Cluster maps categorical data to numerical values. 4. The use of Oracle Data Mining equi-width binning transformation with automated estimation of the required number of bins is highly recommended. 5. The presence of outliers can significantly impact clustering algorithms. Use a clipping transformation before binning or normalizing. Outliers with equi-width binning can prevent O-Cluster from detecting clusters. As a result, the whole population appears to fall within a single cluster.

The `formula` specification has the form `~ terms` where `terms` are the column names to include in the model. Multiple terms are specified using + between column names. Use `~ .` if all columns in `data` should be used for model building. To exclude columns, use − before each column name to exclude.

Value

The function `ore.odmOC` returns an object of class `ore.odmOC`, which includes the following components:

`name`	The name of the in-database model.
`settings`	A `data.frame` of settings used to build the model.
`attributes`	A `data.frame` of attributes used to build the model.
`clusters`	A `data.frame` describing general per-cluster information.
`leaf.cluster.counts`	A `data.frame` containing leaf clusters with support.
`taxonomy`	A `data.frame` describing parent-child cluster relationship.
`centers`	A `data.frame` describing per cluster-attribute centroid information.
`centers2`	A `data.frame` containing center information of leaf clusters.
`histogram`	A `data.frame` describing per cluster-attribute histogram information.

rule	A data.frame descrbing the clustering rules.
formula	The formula used for the model fitted.
call	The matched call.

The function predict returns an ore.frame with columns according to the type and supplemental.cols parameters. If type is "class", the result includes the most likely cluster ID and its probability. If type is "raw", the result includes one column for each cluster ID, and the column values reflect the probability for that cluster ID. Both can be specified together. If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

B.L. Milenova and M.M. Campos, Clustering Large Databases with Numeric and Nominal Values Using Orthogonal Projection, Proceeding of the 29th VLDB Conference, Berlin, Germany (2003).

Oracle9i O-Cluster: Scalable Clustering of Large High Dimensional Data Sets http://www.oracle.com/technology/products/bi/odm/pdf/o_cluster_algorithm.pdf

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmKMeans, ore.odmOC, ore.odmEM, partitions

Examples

```
# a 2-dimensional example
x <- rbind(matrix(rnorm(100, mean = 4, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
X <- ore.push (data.frame(ID=1:100,x))
rownames(X) <- X$ID

oc.mod <- ore.odmOC(~., X, num.centers=2)

summary(oc.mod)
clusterhists(oc.mod)
histogram(oc.mod)
r <- rules(oc.mod)

p <- predict(oc.mod,X)
p <- predict(oc.mod,X,type=c("class","raw"))
p <- predict(oc.mod,X,type=c("class","raw"),supplemental.cols=c("x","y"))
p <- predict(oc.mod,X,type="class")
p <- predict(oc.mod,X,type="class",supplemental.cols=c("x","y"))
p <- predict(oc.mod,X,type="raw")
p <- predict(oc.mod,X,type="raw",supplemental.cols=c("x","y"))
```

`ore.odmRAlg` *Extensible R Algorithm Models*

Description

Creates an Extensible R Algorithm model using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

```
ore.odmRAlg(data,
         mining.function = c("classification", "regression", "clustering",
                  "feature_extraction", "attribute_importance",
                  "association"),
         formula = NULL,
         build.function,
         build.parameter = NULL,
         score.function = NULL,
         detail.function = NULL,
         detail.value = NULL,
         odm.settings = NULL)

## S3 method for class 'ore.odmRAlg'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = "class",
  na.action = na.pass,...)

## S3 method for class 'ore.odmRAlg'
summary(object,...)

## S3 method for class 'summary.ore.odmRAlg'
print(x,...)
```

Arguments

`data`	An `ore.frame` object used for model building.
`mining.function`	A scalar string to specify the type of mining function. The valid values are `classification`, `regression`, `clustering`, `feature_extraction`, `attribute_importance`, `association`.
`formula`	An R formula or a string representing a formula in characters. This formula can be named or take the default name '`formula`'. This name is used to match the argument name of the `build.function` to pass the specified formula. If the formula is `NULL`, the R `build.function` shall not take a formula.
`build.function`	The name of a registered R script used to build the model. The R script uses the first argument for input data, optionally the second argument for weight numeric vector when parameter `odms_row_weight_column_name` is specified in

odm.settings, and matches the rest of the arguments by name with the values from build.parameter. The R script returns an R object representing the fit model. For clustering, the return R object must contain an attribute dm$nclus to specify the number of clusters. For feature extraction, attribute dm$nfeat shall be used to specify the number of features in the object returned.

build.parameter
: A list containing build function parameters excluding input data and weight vector if applicable. The list element names must match with the name of build.function script input parameter names. Only scalar numeric and string are valid parameter values.

score.function
: The name of a registered R script used to score the model. The script takes two arguments for the model and new data respectively and returns a data.frame containing prediction results. For regression, the results are predicted values. In classification, clustering, and feature exaction, the results are probabilities for each class, cluster, and feature respectively. Rows of the results match with rows of input data. For classification, the column names of the results match with the class label names. For clustering and feature extraction, the columns are arranged in the order of the cluster and feature ids.

detail.function
: The name of a registered R script used to obtain model details and return them in an data.frame.

detail.value
: An data.frame object used to specify the data types of the return data.frame from detail.function.

odm.settings
: Same as odm.settings in ore.odmKMeans.

object
: An object of type ore.odmRAlg.

newdata
: Data used for scoring.

supplemental.cols
: Columns to include in the output.

type
: If set to "raw", provides the probability for each class, cluster, or feature returned. If set to "class", the one with the maximum probability is returned. The default is "class".

na.action
: A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

...
: Additional arguments affecting the predictions produced.

x
: An object of type summary.ore.odmRAlg.

Details

Extensible R Algorithm builds, scores, and views R model using the registered R scripts. It supports classification, regression, clustering, feature_extraction, attribute_importance, and association mining function.

Value

An object of class "ore.odmRAlg" includes the following components:

name
: The name of the in-database model.

`mining.function`
: The type of the data mining function for the model.

`details`
: An `ore.frame` returned by the R `detail.function` script.

`settings`
: A `data.frame` of settings used to build the model.

`attributes`
: A `data.frame` of attributes used to build the model.

`formula`
: The `formula` used for the model fitted.

An object of class "summary.ore.odmRAlg" includes the following components:

`name`
: The name of the model object.

`call`
: The function call used to build the model with the given arguments.

`settings`
: A `data.frame` of settings used to build the model.

`details`
: An `ore.frame` returned by the R `detail.function` script.

The `predict` method executes the `score.function` specified for the model build and returns an `ore.frame` containing the predictions along with the columns specified by the `supplemental.cols` argument. Function `predict` is applicable to only `classification`, `regression`, `clustering`, and `feature_extraction` models.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Extensible R Algorithm Model Setting

See Also

`ore.scriptCreate` `ore.scriptDrop` `ore.scriptList` `partitions` `settings`

Examples

```
library(OREembed)

digits <- getOption("digits")
options(digits = 5L)

IRIS <- ore.push(iris)

# Regression with glm
ore.scriptCreate("glm_build", function(data, form, family)
    glm(formula = form, data = data, family = family))

ore.scriptCreate("glm_score", function(mod, data) {
    res <- predict(mod, newdata = data); data.frame(res)})

ore.scriptCreate("glm_detail", function(mod)
    data.frame(name=names(mod$coefficients), coef=mod$coefficients))
```

```
ore.scriptList(name = "glm_build")
ore.scriptList(name = "glm_score")
ore.scriptList(name = "glm_detail")

ralg.glm <- ore.odmRAlg(IRIS, mining.function = "regression",
  formula = c(form="Sepal.Length ~ ."),
  build.function = "glm_build", build.parameter = list(family="gaussian"),
  score.function = "glm_score",
  detail.function = "glm_detail", detail.value = data.frame(name="a", coef=1))

summary(ralg.glm)
predict(ralg.glm, newdata = head(IRIS), supplemental.cols = "Sepal.Length")

ore.scriptDrop(name = "glm_build")
ore.scriptDrop(name = "glm_score")
ore.scriptDrop(name = "glm_detail")

# Classification with nnet
ore.scriptCreate("nnet_build", function(dat, form, sz){
  require(nnet);
  set.seed(1234);
  nnet(formula = formula(form), data=dat,
              size=sz, linout=TRUE, trace=FALSE);
  }, overwrite=TRUE)

ore.scriptCreate("nnet_score", function(mod, data) {
   require(nnet);
   res <- data.frame(predict(mod, newdata = data));
   names(res) <- sort(mod$lev); res})

ore.scriptCreate("nnet_detail", function(mod)
   data.frame(conn=mod$conn, wts=mod$wts),
                overwrite=TRUE)

ralg.nnet <- ore.odmRAlg(IRIS, mining.function = "classification",
  formula = c(form="Species ~ ."),
  build.function = "nnet_build", build.parameter = list(sz=2),
  score.function = "nnet_score",
  detail.function = "nnet_detail", detail.value = data.frame(conn=1, wts =1))

summary(ralg.nnet)
predict(ralg.nnet, newdata = head(IRIS), supplemental.cols = "Species")

ore.scriptDrop(name = "nnet_build")
ore.scriptDrop(name = "nnet_score")
ore.scriptDrop(name = "nnet_detail")

# Feature extraction with pca
ore.scriptCreate("pca_build", function(dat){
  mod <- prcomp(dat, retx = FALSE)
  attr(mod, "dm$nfeat") <- ncol(mod$rotation)
  mod}, overwrite=TRUE)

ore.scriptCreate("pca_score", function(mod, data) {
  res <- predict(mod, data)
  as.data.frame(res)}, overwrite=TRUE)
```

```
ore.scriptCreate("pca_detail", function(mod) {
  rotation_t <- t(mod$rotation)
  data.frame(id = seq_along(rownames(rotation_t)), rotation_t)},
                overwrite=TRUE)

X <- IRIS[, -5L]
ralg.pca <- ore.odmRAlg(X, mining.function = "feature_extraction",
  formula = NULL,
  build.function = "pca_build",
  score.function = "pca_score",
  detail.function = "pca_detail",
  detail.value = data.frame(Feature.ID=1, ore.pull(head(X,1L))))

summary(ralg.pca)
head(cbind(X, Pred = predict(ralg.pca, newdata = X)))

ore.scriptDrop(name = "pca_build")
ore.scriptDrop(name = "pca_score")
ore.scriptDrop(name = "pca_detail")

options(digits = digits)
```

`ore.odmRF` *In-Database Random Forest Models*

Description

Fits a random forest model using Oracle Data Mining,

Usage

```
ore.odmRF(formula,
       data,
       auto.data.prep = TRUE,
       na.action = na.pass,
       odm.settings = NULL,
       ctx.settings = NULL)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
`data`	An `ore.frame` object used for model building.
`auto.data.prep`	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is `TRUE`.
`na.action`	A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.
`odm.settings`	Same as `odm.settings` in `ore.odmRF`.
`ctx.settings`	Text mining settings.

Details

Random Forest is a popular ensemble learning technique for classification and regression. By combining the ideas of bagging and random selection of variables, the algorithm produces a collection of decision trees with controlled variance, while avoiding overfitting - a common problem for decision trees.

The `formula` specification has the form `attribute ~ terms` where `attribute` is the attribute to be used for model build and `terms` is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use `response ~ .` if all columns in `data` should be used for model building. To exclude columns, use − before each column name to exclude.

Value

The `ore.odmRF` function returns an object of class `ore.odmRF`, which includes the following components:

`name`	The name of the in-database model.
`settings`	A `data.frame` of settings used to build the model.
`attributes`	A `data.frame` of variables (attribiutes) used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
`formula`	The `formula` used for the model fitted.
`importance`	Attribute importance of the fitted model
`call`	The matched call.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmSVM`, `ore.odmGLM`, `ore.odmDT`, `partitions`

Examples

```
IRIS <- ore.push(iris)
mod <- ore.odmRF(Species~., IRIS)
mod
summary(mod)
importance(mod)
ans <- predict(mod, IRIS)
head(ans,3)
```

`ore.odmSVD` *In-Database Singular Value Decomposition Models*

Description

Creates a Singular Value Decomposition (SVD) model for feature extraction using Oracle Data Mining in Oracle Database 12.2 or later.

Usage

```
ore.odmSVD(formula,
        data,
        auto.data.prep = TRUE,
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

## S3 method for class 'ore.odmSVD'
features(object,...)

## S3 method for class 'ore.odmSVD'
feature_compare(object,
  newdata,
  compare.cols = NULL,
  supplemental.cols = NULL)

## S3 method for class 'ore.odmSVD'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  topN = NULL,
  topN.attrs = FALSE,...)

u(object)
## S3 method for class 'ore.odmSVD'
u(object)

v(object)
## S3 method for class 'ore.odmSVD'
v(object)

d(object)
## S3 method for class 'ore.odmSVD'
d(object)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. For details of the model specification,

see 'Details'.

`data` An `ore.frame` object used for model building.

`auto.data.prep` A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build.

`na.action` A function to use for handling missing values, either `na.pass` to allow missing values or `na.omit` to remove rows with missing values. The default value is `na.pass`.

`odm.settings` Same as `odm.settings` in `ore.odmKMeans`.

`ctx.settings` Same as `ctx.settings` in `ore.odmKMeans`.

`object` An object of type `ore.odmSVD`.

`newdata` The data used for scoring.

`compare.cols` Columns used to compare the features. All the columns are included by default.

`supplemental.cols` Columns to include in the output.

`type` If set to `"raw"`, provides the probability for each feature returned. If set to `"class"`, the feature with the maximum probability is returned.

`topN` A positive integer that restricts the set of features to those that have one of the specified number of highest probability values.

`topN.attrs` boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

`...` Additional arguments affecting the predictions produced.

Details

Singular Value Decomposition (SVD) is a feature extraction algorithm. SVD is orthogonal linear transformations that capture the underlying variance of the data by decomposing a rectangular matrix into three matrixes: `U`, `D`, and `V`. Matrix `D` is a diagonal matrix and its singular values reflect the amount of data variance captured by the bases.

Value

An object of class "ore.odmSVD" includes the following components:

`name` The name of the in-database model.

`settings` A `data.frame` of settings used to build the model.

`attributes` A `data.frame` of attributes used to build the model.

`formula` The `formula` used for the model fitted.

`call` The matched call.

The `d` function returns an `ore.frame` with the singular values of the input data. Column `FEATURE_ID` represents the singular value ID. The `v` function returns an `ore.frame` whose columns contain the right singular vectors. The column names match with `FEATURE_ID` values. The `u` function returns an `ore.frame` whose columns contain the left singular vectors. `CASE_ID_COLUMN_NAME` and `SVDS_U_MATRIX_OUTPUT` must be specified in argument `odm.settings` to product this result.

The `features` function returns an `ore.frame` the same as the `v` function does except in an (vertical) unpivot format. The columns are:

`FEATURE_ID` The numeric identifier associated with the singular value.

`ATTRIBUTE_NAME`
The name of the attribute.

`ATTRIBUTE_VALUE`
The value of the attribute.

`VALUE` The coefficient value associated with the attribute in the new space.

The `feature_compare` function returns an `ore.frame` containing a `SIMILARITY` column and supplementary columns if specified.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmNMF`, `ore.odmESA`, `partitions`

Examples

```
IRIS <- ore.push(cbind(Id = seq_along(iris[[1L]]), iris))

svd.mod <- ore.odmSVD(~. -Id, IRIS)
summary(svd.mod)
d(svd.mod)
v(svd.mod)
head(predict(svd.mod, IRIS, supplemental.cols = "Id"))

svd.pmod <- ore.odmSVD(~. -Id, IRIS,
              odm.settings = list(odms_partition_columns = "Species"))
summary(svd.pmod)
d(svd.pmod)
v(svd.pmod)
head(predict(svd.pmod, IRIS, supplemental.cols = "Id"))
```

ore.odmSVM *In-Database Support Vector Machine Models*

Description

Fits a support vector machine using Oracle Data Mining to be used for regression, classification, or anomaly detection.

Usage

```
ore.odmSVM(formula,
        data,
        type,
        auto.data.prep = TRUE,
        class.priors = NULL,
        active.learning = TRUE,
        complexity.factor = "system.determined",
        conv.tolerance = 0.0001,
        epsilon = "system.determined",
        cache.size = 50000000,
        kernel.function = "system.determined",
        std.dev = "system.determined",
        outlier.rate = 0.1,
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

## S3 method for class 'ore.odmSVM'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,
  bestN = NULL,
  topN.attrs = FALSE,...)

## S3 method for class 'ore.odmSVM'
coef(object,...)

## S3 method for class 'ore.odmSVM'
print(x,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ore.frame object used for model building.

type — For ore.odmSVM, the type of SVM model: "classification", "regression", or "anomaly.detection". For predict, if set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.

bestN — A positive integer that restricts the returned target classes to the specified number of those that have the highest probability.

topN.attrs — boolean, positive integer, False (default). Returns the top N most influence attributes of the predicted target value for regression if topN_attrs is not False. Returns the top N most influence attributes of the highest probability class for classification if topN_attrs is not False. N is equal to the specified positive integer or 5 if topN_attrs is True.

auto.data.prep — A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.

class.priors — An optional named numerical vector that specifies the priors for the target classes. The names of the vector are the possible target classes. The default value is NULL

active.learning — A logical value that specifies whether to use active learning. The default value is TRUE.

complexity.factor — The complexity factor. The value must be a positive number or "system.determined". The default value is "system.determined".

conv.tolerance — The convergence tolerance. The value must be positive. The default value is 0.001.

epsilon — The regularization setting for regression, similar to the complexity factor. Epsilon specifies the allowable residuals, or noise, in the data. The value must be a positive number or "system.determined". The default is "system.determined".

cache.size — If kernel.function is "gaussian", the setting for the kernel cache size (bytes). The default value is 50,000,000.

kernel.function — The setting for the kernel function. It can be "gaussian", "linear", or "system.determined". The default value is "system.determined".

std.dev — If the kernel.function is "gaussian", the setting for the standard deviation for the SVM Gaussian kernel. The Value must be a positive number or "system.determined". The default value is "system.determined".

outlier.rate — For an anomaly detection model, the setting for desired rate of outliers in the training data. A value in the range of 0 to 1. The default value is 0.1.

na.action — A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.

odm.settings — Same as odm.settings in ore.odmKMeans.

ctx.settings — Same as ctx.settings in ore.odmKMeans.

object — An object of type ore.odmSVM.

x — An object of type ore.odmSVM.

newdata — The data used for prediction.

supplemental.cols
: The columns from newdata to be included as the columns in the ore.frame prediction result.

...
: Additional arguments affecting the predictions produced.

Details

This function invokes the Oracle Data Mining SVM implementation that supports classification, regression, and anomaly detection (one-class classification) with linear or Gaussian kernels and an automatic and efficient estimation of the complexity factor (C) and standard deviation (sigma).

Support Vector Machines (SVMs) for regression utilizes an epsilon-insensitive loss function and works particularly well for high-dimensional noisy data. The implementation also supports "active learning" which forces the SVM algorithm to restrict learning to the most informative training examples and not to attempt to use the entire body of data. In most cases, the resulting models have predictive accuracy comparable to that of a standard (exact) SVM model. Active learning provides a significant improvement in both linear and Gaussian SVM models, whether for classification, regression, or anomaly detection. However, active learning is especially advantageous when using the Gaussian kernel, because nonlinear models can otherwise grow to be very large and can place considerable demands on memory and other system resources.

The SVM algorithm operates natively on numeric attributes. The function automatically "explodes" categorical data into a set of binary attributes, one per category value. For example, a character column for marital status with values married or single would be transformed to two numeric attributes: married and single. The new attributes could have the value 1 (true) or 0 (false). When there are missing values in columns, SVM interprets them as missing at random. The algorithm automatically replaces missing categorical values with the mode and missing numerical values with the mean. SVM requires the normalization of numeric input. Normalization places the values of numeric attributes on the same scale and prevents attributes with a large original scale from biasing the solution. Normalization also minimizes the likelihood of overflows and underflows. Furthermore, normalization brings the numerical attributes to the same scale (0,1) as the exploded categorical data. The SVM algorithm automatically handles missing value treatment and the transformation of categorical data, but normalization and outlier detection must be handled manually.

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. Functions can be applied to response and terms to realize transformations. To exclude columns, use − before each column name to exclude.

The function predict computes predictions based on the input data and model.

Value

The function ore.odmSVM returns an object of class ore.odmSVM including the following components:

name
: The name of the in-database model.

settings
: A data.frame of settings used to build the model.

attributes
: A data.frame of predictor columns used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.

fit.values
: An ore.frame of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns

are 'ACTUAL','PREDICTED','PROBABILITY'. For anomaly detection, the columns are 'PREDICTED' and 'PROBABILITY'.

`residuals` For regression models, an `ore.numeric` vector containing the residual values (PREDICTED - ACTUAL).

`formula` The `formula` used for the symbolic description of the model fitted.

`call` The matched call.

If the model built uses a linear kernel, the following is additionally returned:

`coefficients` The coefficients of the SVM model, one for each predictor variable. If `auto.data.prep` is set to `TRUE`, these coefficients will be in the transformed space (after automatic outlier-aware normalization is applied).

The function `predict` returns an `ore.frame` with columns according to the `type` and `supplemental.cols` parameters. For a classification model, if `type` is `"class"`, the result includes the most likely target class and its probability. If `type` is `"raw"`, the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. For a regression model, a column for the prediction is included in the result with name "PREDICTION". If `supplemental.cols` are specified, the named columns are included in the result. For an anomaly detection model, the result includes a prediction and its probability. If the prediction is 1, the case is considered typical. If the prediction is 0, the case is considered anomalous. This behavior reflects the fact that the model is trained with normal data.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

B. L. Milenova, J. S. Yarmus, and M. M. Campos. SVM in oracle database 10g: removing the barriers to widespread adoption of support vector machines. In Proceedings of the "31st international Conference on Very Large Data Bases" (Trondheim, Norway, August 30 - September 02, 2005). pp1152-1163, ISBN:1-59593-154-6.

Milenova, B.L. Campos, M.M., Mining high-dimensional data for information fusion: a database-centric approach 8th International Conference on Information Fusion, 2005. Publication Date: 25-28 July 2005. ISBN: 0-7803-9286-8. John Shawe-Taylor & Nello Cristianini. Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000.

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

`ore.odmGLM`, `ore.odmNB`, `ore.odmDT`, `partitions`

Examples

```
x <- seq(0.1, 5, by = 0.02)
y <- log(x) + rnorm(x, sd = 0.2)
dat <-ore.push(data.frame(x=x, y=y))
```

```
# Regression
svm.mod <- ore.odmSVM(y~x,dat,"regression",kernel.function="linear")
summary(svm.mod)
coef(svm.mod)
svm.res <- predict(svm.mod,dat,supplemental.cols="x")
head(svm.res,6)

m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl  <- as.factor(m$cyl)
m$vs   <- as.factor(m$vs)
m$ID   <- 1:nrow(m)
MTCARS <- ore.push(m)

# Classification
gears.priors <- c("3" = 0.2, "4" = 0.4, "5" = 0.4)
svm.mod  <- ore.odmSVM(gear ~ .-ID, MTCARS,"classification",
              class.priors=gears.priors)
summary(svm.mod)
coef(svm.mod)
svm.res  <- predict (svm.mod, MTCARS,"gear")
with(svm.res, table(gear,PREDICTION))  # generate confusion matrix

# Anomaly Detection
svm.mod  <- ore.odmSVM(~ .-ID, MTCARS,"anomaly.detection")
summary(svm.mod)
svm.res  <- predict (svm.mod, MTCARS, "ID")
head(svm.res)
table(svm.res$PREDICTION)
```

`ore.odmXGB` *In-Database XGBoost Models*

Description

Fits a XGBoost model using Oracle Data Mining to be used for regression or classification.

Usage

```
ore.odmXGB(formula,
        data,
        type,
        auto.data.prep = TRUE,
        na.action = na.pass,
        odm.settings = NULL,
        ctx.settings = NULL)

## S3 method for class 'ore.odmXGB'
predict(object,
  newdata,
  supplemental.cols = NULL,
  type = c("class","raw"),
  na.action = na.pass,...)
```

```
## S3 method for class 'ore.odmXGB'
importance(object,...)

## S3 method for class 'ore.odmXGB'
print(x,...)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	An ore.frame object used for model building.
type	For ore.odmXGB, the type of XGB model: "classification" or "regression". For predict, if set to "raw", provides probability for each class returned. If set to "class", the class with the maximum probability is returned.
auto.data.prep	A logical value that specifies whether Oracle Data Mining should invoke automatic data preparation for the build. The default value is TRUE.
na.action	A function to use for handling missing values, either na.pass to allow missing values or na.omit to remove rows with missing values. The default value is na.pass.
odm.settings	Same as odm.settings in ore.odmXGB.
ctx.settings	Same as ctx.settings in ore.odmXGB.
object	An object of type ore.odmXGB.
x	An object of type ore.odmXGB.
newdata	The data used for prediction.
supplemental.cols	The columns from newdata to be included as the columns in the ore.frame prediction result.
...	Additional arguments affecting the predictions produced.

Details

This function invokes the Oracle Data Mining XGBoost implementation that supports classification and regression models.

XGBoost is a highly-efficient, scalable gradient tree boosting machine learning algorithm for regression and classification. The XGBoost algorithm prepares training data, builds and persists a model, and applies the model for prediction. It can be used as a stand-alone predictor or be incorporated into real-world production pipelines for a wide range of problems such as ad click-through rate prediction, hazard risk prediction, web text classification, and so on.

The formula specification has the form response ~ terms where response is the numeric or character response vector and terms is a series of terms, for example, the column names, to include in the model. Multiple terms are specified using + between column names. Use response ~ . if all columns in data should be used for model building. Functions can be applied to response and terms to realize transformations. To exclude columns, use − before each column name to exclude.

The function predict computes predictions based on the input data and model.

Value

The function ore.odmXGB returns an object of class ore.odmXGB including the following components:

name	The name of the in-database model.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of predictor columns used to build the model. The columns include: name, type (numerical or categorical), data type, data length (size), precision and scale for numeric data, and whether the variable is the target.
fit.values	An ore.frame of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL','PREDICTED','PROBABILITY'.
residuals	For regression models, an ore.numeric vector containing the residual values (PREDICTED - ACTUAL).
formula	The formula used for the symbolic description of the model fitted.
call	The matched call.
importance	The attribute importance of the XGB model, one for each predictor variable.

The function predict returns an ore.frame with columns according to the type and supplemental.cols paramters. For a classification model, if type is "class", the result includes the most likely target class and its probability. If type is "raw", the result includes one column for each target class and the column values reflect the probability for that class. Both can be specified together. For a regression model, a column for the prediction is included in the result with name "PREDICTION". If supplemental.cols are specified, the named columns are included in the result.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmGLM, ore.odmSVM, ore.odmDT, partitions

Examples

```
x <- seq(0.1, 5, by = 0.02)
y <- log(x) + rnorm(x, sd = 0.2)
dat <-ore.push(data.frame(x=x, y=y))

# Regression
xgb.mod <- ore.odmXGB(y~x,dat,"regression")
summary(xgb.mod)
importance(xgb.mod)
xgb.res <- predict(xgb.mod,dat,supplemental.cols="x")
```

```
head(xgb.res,6)

m <- mtcars
m$gear <- as.factor(m$gear)
m$cyl  <- as.factor(m$cyl)
m$vs   <- as.factor(m$vs)
m$ID   <- 1:nrow(m)
MTCARS <- ore.push(m)

# Classification
xgb.mod  <- ore.odmXGB(gear ~ .-ID, MTCARS,"classification")
summary(xgb.mod)
importance(xgb.mod)
xgb.res  <- predict (xgb.mod, MTCARS,"gear")
with(xgb.res, table(gear,PREDICTION))  # generate confusion matrix
```

`ore.options` *Oracle R Enterprise Global Options*

Description

The global options that affect Oracle R Enterprise operations.

Options for Reporting

`ore.trace`**:** A logical value indicating whether iterative Oracle R Enterprise functions should print output at each iteration. The default value for this option is `FALSE`.

Options for Row Ordering

`ore.sep`**:** A character string specifying the separator to use between multiple column row names of an `ore.frame`. The default value for this option is `"|"`.

`ore.warn.order`**:** A logical value indicating whether a warning should be issued when pulling an `ore.frame` that lacks row names or an `ore.vector` that lacks element names into memory. The default value for this option is `TRUE`.

Options for Server Execution

`ore.parallel`**:** A preferred degree of parallelism to use in the embedded R job; either a positive integer greater than or equal to `2` for a specific degree of parallelism, a value of `FALSE` or `1` for no parallelism, a value of `TRUE` for the database's default for parallelism, or `NULL` for the database default for the operation. The default value for this option is `NULL`.

Options for Subsetting

`ore.na.extract`**:** A logical value used during logical subscripting of an `ore.frame` or `ore.vector` object. When `TRUE` rows or elements with an `NA` logical subscript produces rows or elements with `NA` values. When `FALSE` an `NA` logical subscript is interpreted as a `FALSE` value, resulting in the removal of the corresponding row or element. The default value for this option is `FALSE`, whereas a value of `TRUE` would mimic how R treats missing value logical subscripting of `data.frame` and `vector` objects.

Options for Serialization

`ore.envAsEmptyenv`: A logical value indicating whether referenced environments in an object should be replaced with an empty environment during serialization to an Oracle Database. When `TRUE`, the referenced environment in the object will be replaced with an empty environment whose parent is `.GlobalEnv`, and therefore, the objects in the original referenced environment will not be serialized. In some situations, this could significantly reduce the size of serialized objects. When `FALSE`, all the objects in the referenced environment will be serialized, and could be unserialized and loaded into memory. The default value for this option is `FALSE`.

Serialization is used across Oracle R Enterprise. It is used in `ore.save` before saving objects to a datastore. It is used in the method `ore.push` for a `list`, saving a serialized `list` object to the database. It is used as well in the embedded R functions for serializing parameters of `list` type and serializing some objects returned by an embedded R function.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`options`, `ore.save`, `ore.datastore`, `ore.doEval`

Examples

```
options()[c("ore.envAsEmptyenv", "ore.na.extract", "ore.parallel", "ore.sep",
           "ore.trace", "ore.warn.order")]
```

---

ore.predict *Oracle R Enterprise Predictions Using R Models*

Description

Generic for model predictions in Oracle R Enterprise.

Usage

```
ore.predict(object, newdata, ...)
```

Arguments

`object`	A model object.
`newdata`	An `ore.frame` object.
`...`	Optional arguments for implemented methods.

Value

Returns an object of an `ore` subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict-lm, ore.predict-glm, ore.predict-kmeans, ore.predict-matrix, ore.predict-multinom, ore.predict-nnet, ore.predict-ore.model, ore.predict-rpart.

---

ore.predict-glm *Oracle R Enterprise Predictions Using* glm *Models*

Description

Oracle R Enterprise method for generating predictions using glm models.

Usage

```
  ## S4 method for signature 'glm'
ore.predict(object, newdata, type = c("link", "response"),
            se.fit = FALSE, dispersion = NULL, na.action = na.pass,
            ...)
```

Arguments

object	A glm model object.
newdata	An ore.frame object.
type	A character string specifying the type of prediction to make; either "link" (scale of the link function) or "response" (scale of the response variable).
se.fit	A logical indicating whether to return the standard errors for the predictions.
dispersion	The dispersion parameter to use when calculating the standard errors for the predictions.
na.action	The manner in which NA values are handled, either na.omit or na.pass.
...	Optional arguments.

Value

When argument se.fit is FALSE, returns an ore.numeric object containing the predictions in the specified type.

When argument se.fit is TRUE, returns an ore.frame object with two columns: "PRED" and "SE.PRED".

Note

Use of date/time terms in this method will result in an error.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, ore.predict-lm, predict.glm.

Examples

```
infertModel <-
  glm(case ~ age + parity + education + spontaneous + induced,
      data = infert, family = binomial())
INFERT <- ore.push(infert)
INFERTpred <- ore.predict(infertModel, INFERT, type = "response",
                          se.fit = TRUE)
INFERT <- cbind(INFERT, INFERTpred)
head(INFERT)
```

---

ore.predict-kmeans *Oracle R Enterprise Predictions Using* kmeans *Models*

Description

Oracle R Enterprise method for generating predictions using kmeans Models.

Usage

```
  ## S4 method for signature 'kmeans'
ore.predict(object, newdata, type = c("classes", "distances"),
            na.action = na.pass, ...)
```

Arguments

object	A kmeans model object.
newdata	An ore.frame object.
type	A character string specifying the type of prediction to make; either "classes" (cluster id) or "distances" (Euclidean distance from cluster centers).
na.action	The manner in which NA values are handled, either na.omit or na.pass.
...	Optional arguments.

Value

If argument type is "classes", returns an ore.integer object of cluster classifications.

If argument type is "distances", returns an ore.frame object with one column for each cluster.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, ore.predict-matrix, kmeans.

Examples

```
irisClusters <- kmeans(as.matrix(iris[1:4]), centers = 3)
IRIS <- ore.push(iris)
IRIS$CLUSTER <- ore.predict(irisClusters, IRIS)
IRIS <- cbind(IRIS, ore.predict(irisClusters, IRIS, type = "distances"))
head(IRIS)
table(IRIS$CLUSTER, IRIS$Species)
```

---

ore.predict-lm *Oracle R Enterprise Predictions Using* lm *Models*

Description

Oracle R Enterprise method for generating predictions using lm models.

Usage

```
  ## S4 method for signature 'lm'
ore.predict(object, newdata, se.fit = FALSE, scale = NULL, df = Inf,
            interval = c("none", "confidence", "prediction"),
            level = 0.95, na.action = na.pass, pred.var = NULL,
            weights = NULL, ...)
```

Arguments

object	An lm model object.
newdata	An ore.frame object.
se.fit	A logical indicating whether to return the standard errors for the predictions.
scale	The scale parameter for standard error of the predictions.
df	The degrees of freedom for the predictions when argument scale is not NULL.
interval	The type of interval to return, either "none", "confidence", or "prediction".
level	The level for argument interval.
na.action	The manner in which NA values are handled, either na.omit or na.pass.
pred.var	When argument interval is "prediction", the variance for a single observation.

weights	When argument interval is "prediction" and argument pred.val is NULL and object$weights is not NULL, the variance weights for the predictions as either an ore.numeric object or a one-sided model formula referring to data within argument newdata.
...	Optional arguments.

Value

When argument se.fit is FALSE and argument interval is "none", returns an ore.numeric object containing the predictions.

Otherwise returns an ore.frame object with up to four columns: "PRED", "SE.PRED" (when argument se.fit is TRUE), "LOWER.CONF" and "UPPER.CONF" (when argument interval is "confidence"), and "LOWER.PRED" and "UPPER.PRED" (when argument interval is "prediction").

Note

Use of date/time terms in this method will result in an error.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, ore.predict-glm, predict.lm.

Examples

```
irisModel <- lm(Sepal.Length ~ ., data = iris)
IRIS <- ore.push(iris)
IRISpred <- ore.predict(irisModel, IRIS, se.fit = TRUE,
                        interval = "prediction")
IRIS <- cbind(IRIS, IRISpred)
head(IRIS)
```

---

ore.predict-matrix *Oracle R Enterprise Predictions Using Distances to Rows in a Matrix*

Description

Oracle R Enterprise method for generating predictions using rows of a data matrix. For each row in argument newdata, the distances to each row in argument object are computed and either those distances or the row number from argument object with the minimum distance is returned.

Usage

```
  ## S4 method for signature 'matrix'
ore.predict(object, newdata, type = c("classes", "distances"),
            method = "euclidean", p = 2, na.action = na.pass, ...)
```

Arguments

object	A matrix object with no more than 1000 rows.
newdata	An ore.frame object.
type	A character string specifying the type of prediction to make; either "classes" (row id) or "distances".
method	A character string specifying the distance measure to use; either "euclidean", "maximum", "manhattan", "canberra", or "minkowski". See function dist for further explanations.
p	The power of the Minkowski distance when argument method is "minkowski".
na.action	The manner in which NA values are handled, either na.omit or na.pass.
...	Optional arguments.

Value

If argument type is "classes", returns an ore.integer object of row number references to argument object.

If argument type is "distances", returns an ore.frame object with one column for each row in argument object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, ore.predict-kmeans, dist.

Examples

```
groups <- cutree(hclust(dist(iris[1:4], "manhattan")), 3)
centers <- do.call(rbind, lapply(split(iris[1:4], groups), colMeans))
rownames(centers) <- sprintf("DISTANCE%d", 1:3)
IRIS <- ore.push(iris)
IRIS$CLUSTER <- ore.predict(centers, IRIS, method = "manhattan")
IRIS <- cbind(IRIS, ore.predict(centers, IRIS, type = "distances",
                                method = "manhattan"))
head(IRIS)
table(IRIS$CLUSTER, IRIS$Species)
```

ore.predict-multinom

Oracle R Enterprise Predictions Using multinom *Models*

Description

Oracle R Enterprise method for generating predictions using multinom models.

Usage

```
  ## S4 method for signature 'multinom'
ore.predict(object, newdata, type = c("class", "probs"),
            na.action = na.pass, ...)
```

Arguments

object	An multinom model object.
newdata	An ore.frame object.
type	A character string specifying the type of prediction to make; either "class" or "probs".
na.action	The manner in which NA values are handled, either na.omit or na.pass.
...	Optional arguments.

Value

Returns an object of an ore subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, predict.multinom.

Examples

```
library(nnet)
model <- multinom(Species ~ ., data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(model, IRIS, type = "probs"))
head(IRIS)
```

`ore.predict-nnet` *Oracle R Enterprise Predictions Using* `nnet` *Models*

Description

Oracle R Enterprise method for generating predictions using `nnet` models.

Usage

```
  ## S4 method for signature 'nnet.formula'
ore.predict(object, newdata, type = c("raw", "class"),
            na.action = na.pass, ...)
```

Arguments

`object`	An `nnet` model object.
`newdata`	An `ore.frame` object.
`type`	A character string specifying the type of prediction to make; either `"raw"` or `"class"`.
`na.action`	The manner in which `NA` values are handled, either `na.omit` or `na.pass`.
`...`	Optional arguments.

Value

Returns an object of an `ore` subclass.

Author(s)

Oracle `<oracle-r-enterprise@oracle.com>`

References

Oracle R Enterprise

See Also

`ore.predict`, `predict.nnet`.

Examples

```
library(nnet)
model <- nnet(Species ~ ., data = iris, size = 2, rang = 0.1,
              decay = 5e-4, maxit = 200, trace = FALSE)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(model, IRIS))
```

`ore.predict-ore.model`
Oracle R Enterprise Predictions Using `ore.model` *Models*

Description

Oracle R Enterprise method for generating predictions using `ore.model` models.

Usage

```
  ## S4 method for signature 'ore.model'
ore.predict(object, newdata, ...)
```

Arguments

`object`	An `ore.model` model object.
`newdata`	An `ore.frame` object.
`...`	Optional arguments for implemented methods.

Value

Returns an object of an `ore` subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict`

Examples

```
library(OREmodels)
IRIS <- ore.push(iris)
IRISModel <- ore.lm(Sepal.Length ~ ., data = IRIS)
IRIS$PRED <- ore.predict(IRISModel, IRIS)
head(IRIS)
```

---

ore.predict-prcomp *Oracle R Enterprise Predictions Using* prcomp *Models*

Description

Oracle R Enterprise method for generating predictions using prcomp models.

Usage

```
  ## S4 method for signature 'prcomp'
ore.predict(object, newdata, ...)
```

Arguments

object	A prcomp object.
newdata	An ore.frame object.
...	Optional arguments.

Value

Returns an ore.frame object containing the rotated columns of newdata.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.predict, prcomp.

Examples

```
irisModel <- prcomp(~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
                    data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(irisModel, IRIS))
```

`ore.predict-princomp`
Oracle R Enterprise Predictions Using `princomp` *Models*

Description

Oracle R Enterprise method for generating predictions using `princomp` models.

Usage

```
  ## S4 method for signature 'princomp'
ore.predict(object, newdata, ...)
```

Arguments

`object`	A `princomp` object.
`newdata`	An `ore.frame` object.
`...`	Optional arguments.

Value

Returns an `ore.frame` object containing the rotated columns of `newdata`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict`, `princomp`.

Examples

```
irisModel <- princomp(~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
                      data = iris)
IRIS <- ore.push(iris)
IRIS <- cbind(IRIS, ore.predict(irisModel, IRIS))
```

`ore.predict-rpart` *Oracle R Enterprise Predictions Using* `rpart` *Models*

Description

Oracle R Enterprise method for generating predictions using `rpart` models.

Usage

```
  ## S4 method for signature 'rpart'
ore.predict(object, newdata,
            type = c("vector", "prob", "class", "matrix"),
            na.action = na.pass, ...)
```

Arguments

`object`	An `rpart` model object.
`newdata`	An `ore.frame` object.
`type`	A character string specifying the type of prediction to make; either `"vector"`, `"prob"`, `"class"`, or `"matrix"`.
`na.action`	The manner in which `NA` values are handled, either `na.omit` or `na.pass`.
`...`	Optional arguments.

Value

Returns an object of an `ore` subclass.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.predict`, `predict.rpart`.

Examples

```
library(rpart)
model <- rpart(Kyphosis ~ ., data = kyphosis)
KYPHOSIS <- ore.push(kyphosis)
KYPHOSIS <- cbind(KYPHOSIS, ore.predict(model, KYPHOSIS))
head(KYPHOSIS)
```

ore.pull *Oracle R Enterprise Data Exchange Functions*

Description

Pulls or pushes data between an R session and an Oracle R Enterprise schema.

Usage

```
ore.pull(x, ...)
ore.push(x, ...)
```

Arguments

x	For function ore.pull, ore objects. For function ore.push, vector, data.frame, matrix, or list objects.
...	For future expansion.

Details

Functions ore.push and ore.pull place data in or retrieve data from an Oracle R Enterprise schema. The supported in-memory R object to ore object mappings are as follows:

- logical maps to ore.logical
- integer maps to ore.integer
- numeric maps to ore.numeric
- character maps to ore.character
- factor maps to ore.factor
- raw maps to ore.raw
- Date maps to ore.date
- POSIXt maps to ore.datetime
- difftime maps to ore.difftime
- data.frame with columns of type logical, integer, numeric, character, factor, list of raw, Date, POSIXct, difftime maps to ore.frame with the appropriately typed columns

In addition to the mappings above, for function ore.push if the input is a list object, ore.push is applied recursively to each element. Meanwhile, the list elements are serialized before saving to the database. The argument envAsEmptyenv in ... is a logical value indicating whether referenced environments in the list element objects to be saved should be replaced with an empty environment during serialization. When TRUE, the referenced environment will be replaced with an empty environment whose parent is .GlobalEnv, and therefore, the contents in the referenced environment will not be serialized and saved to the database. In some situations, this could significantly reduce the size of the saved objects. When FALSE, the contents in the referenced environment will be serialized and saved, and could be unserialized and loaded into memory when calling ore.pull. The default value is regulated by the global option ore.envAsEmptyenv.

For in-memory R types listed in the table above, the ore.push function creates objects that maintain vector element or data set row ordering of the original data object. For all other data types the behavior of ore.push is unspecified.

Attribute ora.type can be used to specify mapping R types character and factor to CLOB and R type raw to BLOB. The ora.type takes the string value "clob" and "blob" respectively for these two cases.

Value

Function ore.pull returns an in-memory R object containing the appropriate in-memory data.

Function ore.push returns an ore object of the appropriate type.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.ls, ore.frame, ore.matrix, ore.vector ore.options

Examples

```
vec <- 1:10
oreVec <- ore.push(vec)
class(oreVec)
vec2 <- ore.pull(oreVec)
class(vec2)
oreVec3 <- ore.push(oreVec)
class(oreVec3)
vec
oreVec
vec2
oreVec3

IRIS <- ore.push(iris)
class(IRIS)
new.iris <- ore.pull(IRIS)
class(new.iris)
head(IRIS)
head(new.iris)

vraw <- raw(2000L)
oreRaw <- ore.push(vraw)
class(oreRaw)
new.vraw <- ore.pull(oreRaw)
class(new.vraw)
length(new.vraw)

vbraw <- raw(3000L)
attr(vbraw, "ora.type") <- "blob"
oreBRaw <- ore.push(vbraw)
class(oreBRaw)
new.vbraw <- ore.pull(oreBRaw)
class(new.vbraw)
length(new.vbraw)
```

```
attr(iris$Species, "ora.type") <- "clob"
iris$Species.raw <- lapply(iris$Species,
                           function(x) charToRaw(as.character(x)))
attr(iris$Species.raw, "ora.type") <- "blob"
IRIS2 <- ore.push(iris)
class(IRIS2)
new.iris2 <- ore.pull(IRIS2)
class(new.iris2)
```

`ore.rank` *Oracle R Enterprise Data Ranking*

Description

Enables the investigation of the distribution of values along numeric columns in an `ore.frame` object. Highlights include: Allows ranking within groups, Partitions observations into groups based on rank tiles, Provides options for treatment of ties, Calculates cumulative percentages and percentiles, Calculates normal scores from ranks.

Usage

```
ore.rank(data, var, desc = FALSE, groups = NULL, group.by = NULL,
         ties = c("mean", "high", "low", "dense",
                  "condense"),
         score = c("none", "fraction", "nplus1", "blom", "tukey",
                   "vw", "percent", "savage",
                   "waerden", "fn1", "n1"),
         fraction = FALSE, percent = FALSE, nplus1 = FALSE,
         savage = FALSE, blom = FALSE, tukey = FALSE, vw = FALSE)
```

Arguments

`data`	An `ore.frame` object.
`var`	A comma-separated character string specifying the names of numeric columns within argument `data`.
`desc`	A logical value indicating whether to rank in ascending or descending order.
`groups`	An optional numeric value specifying the number of partitions in the data. For percentiles, specify `groups = 100`. For deciles, specify `groups = 10`. For quartiles, specify `groups = 4`.
`group.by`	An optional character vector specifying the group by column names within argument `data`.
`ties`	A character string specifying how to handle ties; One of `"low"` (smallest rank within the tied group), `"high"` (largest rank within the tied group), `"mean"` (average rank within the tied group), and `"dense"/"condense"` (arbitrary unique rank within the tied group).
`score`	A character string specifying a score; One of `"none"`, `"fraction"`, `"nplus1"`, `"blom"`, `"tukey"`, `"vw"`, `"percent"`, or `"savage"`, `"waerden"`, `"fn1"`, or `"n1"`.

fraction	A logical value indicating whether to compute the ratio of 'rank/#non-missing values' for each column in argument var.
percent	A logical value indicating whether to compute the ratio '(rank * 100)/#non-missing values' for each column in argument var.
nplus1	A logical value indicating whether to compute the ratio 'rank/(#non-missing values + 1)' for each column in argument var.
savage	Equivalent to score = "savage".
blom	Equivalent to score = "blom".
tukey	Equivalent to score = "tukey".
vw	Equivalent to score = "vw".

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.sort

Examples

```
IRIS <- ore.push(iris)

# Rank 2 columns with column aliases and sort them in descending rank order
ore.rank(data = IRIS,
         var  = "Petal.Length=Col1Rank, Sepal.Length=Col2Rank",
         desc = TRUE)

# Handling of ties
ore.rank(data = IRIS,
         var  = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
         ties = "low")

# Rank within each Species group
ore.rank(data = IRIS,
         var  = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
         group.by = "Species")

# Partition rows into 10 groups to get deciles
ore.rank(data = IRIS,
         var  = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
         groups = 10)

# Estimate the cumulative distribution function
ore.rank(data = IRIS,
         var  = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
```

```
            nplus1 = TRUE)

    # Calculate scores
    ore.rank(data = IRIS,
            var = "Petal.Length=PetalLengthRanks, Sepal.Length=SepalRanks",
            score = "savage", groups = 100, group.by = "Species")
```

`ore.raw-class` *Class* `ore.raw`

Description

The `ore.raw` class represents `raw` data columns in Oracle R Enterprise.

Raw Data Methods

ore.pull `signature(x = "ore.raw")`: Returns an R `raw` object that contains the binary fetched from `x`

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore`, `ore.datetime`, `ore.character`, `ore.factor`, `ore.frame`, `ore.logical`, `ore.matrix`, `ore.number`, `ore.vector`

Examples

```
val <- ore.push(charToRaw('ABCDE'))
class(val)              # ore.raw

val2 <- ore.pull(val)
class(val2)             # raw
```

ore.recode *Oracle R Enterprise ore.vector Value Recode Function*

Description

Recodes the values in an ore.vector object.

Usage

```
ore.recode(x, old, new, default = NULL)
```

Arguments

x	An ore.vector object.
old	An R vector specifying the old values in x.
new	Either an R vector of the same length as argument old or an R matrix with the same number of rows as argument old specifying the new values.
default	A single value to use for the non-matched elements in argument old. If NULL, non-matched elements are converted to NA values.

Value

If argument new is an R vector, returns an ore.vector object representing a recoded vector.

If argument new is a R matrix, returns an ore.frame object where each column of argument new generates a corresponding recoded column in the return value.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.vector, ore.hash

Examples

```
vec <- ore.push(c("a", "b", NA, "a", "c", NA, "e"))
ore.recode(vec, c("a", "b", "c"), c("able", "baker", "charlie"))
ore.recode(vec, c("a", "b", "c"),
           cbind(lower = c("able", "baker", "charlie"),
                 UPPER = c("ABLE", "BAKER", "CHARLIE")))
```

ore.rm *Oracle R Enterprise Object Removal Function*

Description

Removes ore.frame objects, representing database tables and views, from the R environment for a schema in the Oracle R Enterprise session. Corresponding data within the database are not affected.

Usage

```
ore.rm(list = character(0L), schema)
```

Arguments

list	A character vector naming ore.frame objects to be removed from the attached schema.
schema	A character string specifying the database schema name.

Details

Function ore.rm removes the specified objects in argument list from the specified argument schema. If argument schema is unspecified, the default schema - the one specified at connection time for the Oracle R Enterprise session - is used.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.attach, ore.connect, ore.exists, ore.get, ore.ls, ore.sync

Examples

```
if (!interactive())
{
  ore.create(iris, table="IRIS_TABLE")
  ore.exists("IRIS_TABLE")
  ore.rm("IRIS_TABLE")
  ore.exists("IRIS_TABLE")
  ore.sync()
  ore.exists("IRIS_TABLE")

  ore.drop("IRIS_TABLE")    # clean up
}
```

ore.roll *Oracle R Enterprise Rolling Window Aggregates*

Description

Performs rolling univariate aggregation methods for ore.number vectors.

Usage

```
ore.rollmax(x, k, align = c("center", "left", "right"),
            na.rm = FALSE, ...)

ore.rollmin(x, k, align = c("center", "left", "right"),
            na.rm = FALSE, ...)

ore.rollsum(x, k, align = c("center", "left", "right"),
            na.rm = FALSE, ...)

ore.rollmean(x, k, align = c("center", "left", "right"),
             na.rm = FALSE, ...)

ore.rollsd(x, k, align = c("center", "left", "right"),
           na.rm = FALSE, ...)

ore.rollvar(x, k, align = c("center", "left", "right"),
            na.rm = FALSE, ...)
```

Arguments

x	An ordered ore.vector object for ore.rollmax and ore.rollmin; an ordered ore.number object otherwise.
k	A positive integer width for the rolling window.
align	A character string specifying the type of alignment; either "center", "left", or "right" justified.
na.rm	A logical value indicating whether NA values should be ignored during the calculations.
...	Additional arguments.

Details

These rolling window aggregate functions take an ordered ore.vector or ore.number object and create moving windows using one of the the following align types:

"center" ceiling((k -1)/2) preceding rows, the current row, and floor((k -1)/2) following rows

"left" the current row and the k -1 rows that follow

"right" the current row and the k -1 preceding rows

These moving windows produce smaller widths at the beginning and ending of the series. For example if argument k is 5 and argument align is "center", then the first aggregate uses subset x[1:3], the second aggregate subset x[1:4], and the third aggregate subset x[1:5].

Value

For ore.rollmax and ore.rollmin, returns an ore.vector object of the same type as argument x. Otherwise returns an ore.numeric object. The output object always has the same length as argument x.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.number-univar, ore.vector-aggregate

Examples

```
NHTEMP <- ore.frame(year = 1912:1971, temp = as.vector(nhtemp))
rownames(NHTEMP) <- NHTEMP$year
NHTEMP$rollmean7 <- ore.rollmean(NHTEMP$temp, k = 7)
NHTEMP$rollsd7 <- ore.rollsd(NHTEMP$temp, k = 7)
head(NHTEMP, 10)
```

---

ore.rules-class *Class* ore.rules

Description

The ore.rules class represents a set of rules of an ore.odmAssocRules object in an Oracle Database instance. An ore.odmAssocRules object represents an Association model created by Oracle Data Mining.

Subsetting

In the code snippets below, argument x is an ore.rules object.

subset(x,topN=NULL,min.support=NUll,min.confidence=NULL,min.lift=NULL,max.rule.

Returns a new ore.rules object using:

topN The top number of rules to return. If topN has been specified in the object x, the subset method does not work for this object, and will report an error.

min.support The minimum support of the rules to return.

min.confidence The minimum confidence of the rules to return.

min.lift The minimum lift the rules to return.

max.rule.length The maximum length of the rules to return. The length of a rule is defined as the total number of the items in a rule, including both lhs and rhs.

min.rule.length The minimum length of the rules to return.

rule.id The rule ID (numeric type) of the rule to return. If a rule.id is specified, the other filtering parameters are ignored.

orderby A vector of column names to be ordered. The vector should be a subset of the set ["confidence", "support", "lift", "number_of_items"]. By default, the rules are sorted by confidence in descending order and then by support in descending order.

decreasing A vector of logical variables that indicates whether the corresponding column of the orderby vector is sorted in descending order. The length of this vector should be the same as the length of orderby, with an exception of length 1, which indicates that all the columns in orderby are sorted in the same order.

lhs A list of items. This method returns all rules in which the antecedent contains one or more of the specified items. For example, lhs=list("apple","orange") or lhs=list(age=31,occupation="engineer").

rhs A list of items. This method returns all rules in which the consequent contains one or more of the specified items.

Utilities

In the code snippets below, argument x is an ore.rules object.

ore.pull(x): Returns the in-memory counterpart object of the ore.rules object, the rules object defined in the **arules** package.

Details

The ore.rules class represents a set of rules in an Oracle Database. The ore.rules object is the result of an Association Rules model.

The column names of the ore.rules object are "RULE_ID", "NUMBER_OF_ITEMS", "LHS", "RHS", "SUPPORT", "CONFIDENCE", "LIFT".

You can pull rules into memory in a local R session from an Oracle Database instance by using ore.pull. The in-memory object is of class rules, which is defined in the **arules** package.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.itemsets, ore.odmAssocRules, rules

Examples

```
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
  item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
  max.rule.length = 3)
```

```
rules <- rules(ar.mod)

sub.rules1 <- subset(rules, min.confidence=0.7, lhs=list("b", "c"))
sub.rules2 <- subset(sub.rules1, max.rule.length=2)

library(arules, warn.conflicts=FALSE)
rules.arules <- ore.pull(rules)
inspect(rules.arules[1:3])
rules.arules1 <- ore.pull(sub.rules1)
rules.arules2 <- ore.pull(sub.rules2)
```

`ore.save` *Oracle R Enterprise Datastore Saving Function*

Description

Saves R objects into a datastore in the user's Oracle Database schema.

Usage

```
ore.save(..., list = character(0),
         name = stop("parameter 'name' must be specified"),
         grantable = FALSE,
         envir = parent.frame(), overwrite = FALSE, append = FALSE,
         description = character(0),
         envAsEmptyenv = getOption("ore.envAsEmptyenv", FALSE))
```

Arguments

`...`	The names of the R objects to be saved (as symbols or character strings).
`list`	A character vector containing the names of R objects to be saved.
`name`	A character string specifying the name of datastore in which to save the R objects.
`grantable`	A scalar logical value specifying whether to create a new datastore the read privilege for which can be granted to other users. Argument `grantable` is ignored when used with argument `overwrite` or `append`.
`envir`	An environment to search for the R objects to be saved.
`overwrite`	A logical value specifying whether to overwrite the datastore, if it already exists.
`append`	A logical value specifying whether to append the R objects to the datastore, if it already exists.
`description`	A character string containing no more than 2000 characters to be used as comments for the datastore.
`envAsEmptyenv`	A logical value indicating whether referenced environments in the R objects to be saved should be replaced with an empty environment during serialization. When `TRUE`, the referenced environment in the objects will be replaced with an empty environment whose parent is `.GlobalEnv`, and therefore, the contents in the original referenced environment will not be serialized and saved to the database. In some situations, this could significantly reduce the size of

the saved objects. When FALSE, the contents in the referenced environment will be serialized and saved, and could be unserialized and loaded into memory when calling ore.load. The default value is regulated by the global option ore.envAsEmptyenv.

Details

Function ore.save saves the R objects from argument ... or itemized in argument list for retrieval from environment envir to the datastore specified in argument name of the user's Oracle Database schema. Nonexistent R objects referenced in argument list are ignored. If no R objects given in argument list exist in environment envir, the function will throw an error.

By default, R objects are saved to a new datastore with the specified name. Arguments overwrite and append can be used to save R objects to an existing datastore, but only one of them can be TRUE.

Comments can be added to datastore through argument description.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.load, ore.move, ore.delete, ore.datastore, ore.datastoreSummary ore.options

Examples

```
if (any(sapply(c("x", "y", "z"), exists)))
  stop("object x, y, or z exists")

x <- stats::runif(20)
y <- list(a = 1, b = TRUE, c = "oops")
z <- ore.push(x)

# save all objects in the current workspace environment to
# a datastore with name 'rq$ds_1' in the user's schema
ore.save(list=ls(), name="rq$ds_1")

# overwrite existing datastore rq$ds_1 with object x, y in
# the current workspace environment
ore.save(x, y, name="rq$ds_1", overwrite=TRUE)

# add object z in the current workspace environment to
# the existing datastore rq$ds_1
ore.save(z, name="rq$ds_1", append=TRUE)

ore.delete(name="rq$ds_1")
rm(x, y, z)
```

`ore.scriptCreate` *Oracle R Enterprise Embedded R Script Creation, List, Load, Drop Functions*

Description

Creates an R script, which contains a single function definition, in the Oracle Database R script repository, or lists, loads, or drops an R script from the repository.

Usage

```
ore.scriptCreate(name, FUN, global = FALSE, overwrite = FALSE)
ore.scriptList(name = NULL, pattern = NULL,
               type = c("user", "global", "system", "all", "grant", "granted"))
ore.scriptLoad(name = NULL, pattern = NULL, owner = NULL, newname = NULL, envir
ore.scriptDrop(name = NULL, pattern = NULL, global = FALSE, silent = FALSE)
```

Arguments

`name`	A character string that specifies the name of an R script in the R script repository; cannot be used with argument `pattern`.
`FUN`	A function definition to be used with functions `ore.doEval`, `ore.groupApply`, `ore.indexApply`, `ore.rowApply`, or `ore.tableApply`. The function cannot recursively call these embedded R APIs.
`pattern`	An optional regular expression character string specifying the matching R script names; cannot be used with argument `name`.
`global`	An optional logical value indicating whether to create or drop a global R script. The default value is `FALSE`, which indicates the R script to create or drop is a private script. Every user has read access to a global R script while only the user who created the R script can access and drop a private R script. Functions `ore.grant` and `ore.revoke` can be used to grant or revoke the read privilege for a private R script to other users.
`overwrite`	A logical value specifying whether to overwrite the named R script, if it already exists.
`type`	A scalar character string specifying the type of R script to list. The valid value is 'user' (default), 'all', 'grant', 'granted', or 'global'. 'user' lists R scripts created by current session user. 'grant' lists R scripts the read privilege for which has been granted by the current session user to other users. 'granted' lists R scripts the read privilege for which has been granted by other users to the current session user. 'global' lists all user-created global R scripts. 'system' lists all system predefined R scripts. 'all' lists all R scripts to which the current session user has read access.
`owner`	An optional character string specifying the user who created the named R script. Argument `owner` can be used along with argument `name` to specify which R script to load into an R environment. Without the `owner` argument, `ore.scriptLoad` finds and loads the R script that matches `name` in the following order: R script that the current session user created, global R script.

newname	An optional character string specifying a new name in the current session for the R function that is loaded by ore.scriptLoad. The newname value must be a valid R function name. By default, the name of the R function takes the name of the script. Cannot be used with argument pattern.
envir	An optional R environment variable that specifies the environment that contains the R function into which ore.scriptLoad loads the named R script.
silent	An optional logical value indicating whether to display an error message when ore.scriptDrop encounters an error in dropping the named R script.

Details

Functions ore.scriptCreate, ore.scriptDrop require the user to have the 'RQADMIN' Oracle Database role.

Value

Function ore.scriptCreate returns an invisible NULL value if it succeeds in creating an R script; otherwise it produces an error. Function ore.scriptLoad returns an invisible NULL value if argument name is used and it succeeds in loading an R script, or, if argument pattern is used, it returns a list of script names that have been successfully loaded; otherwise it produces an error or a warning message. Function ore.scriptDrop returns an invisible NULL value if argument name is used and it succeeds in dropping an R script, or if argument pattern is used, it returns a list of script names that have been successfully dropped; otherwise it produces an error. Function ore.scriptList returns a data frame that contains the columns NAME and SCRIPT, and optionally the columns OWNER and GRANTEE.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.doEval

Examples

```
if (!interactive())
{

  # create an R script for the current user
  ore.scriptCreate("MYLM",
                   function(data, formula, ...) lm(formula, data, ...))
  IRIS <- ore.push(iris)
  ore.tableApply(IRIS[1:4], FUN.NAME = "MYLM",
                 formula = Sepal.Length ~ .)

  # create a global R script available to any user
  ore.scriptCreate("GLBGLM",
                   function(data, formula, ...)
                     glm(formula=formula, data=data, ...),
```

```
                    global = TRUE)
  ore.tableApply(IRIS[1:4], FUN.NAME = "GLBGLM",
                 formula = Sepal.Length ~ .)

  # list R scripts
  ore.scriptList()
  ore.scriptList(pattern="LM", type="all")

  # load an R script to an R function object
  ore.scriptLoad(name="MYLM")
  ore.scriptLoad(name="GLBGLM", newname="MYGLM")
  MYLM(iris, formula = Sepal.Length ~ .)
  MYGLM(iris, formula = Sepal.Length ~ .)

  # grant and revoke R script read privilege to and from public
  ore.grant(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")
  ore.revoke(name = "MYLM", type = "rqscript")
  ore.scriptList(type="grant")

  # drop an R script
  ore.scriptDrop("MYLM")
  ore.scriptDrop("GLBGLM", global=TRUE)

  ore.scriptList(type="all")
}
```

`ore.showHiveOptions`
Show HIVE Options

Description

Shows the current value of all HIVE options, namely, field delimiters for the HIVE tables and the database name.

Usage

```
ore.showHiveOptions()
```

Details

This function displays the values of HIVE options set in the current environment. The HIVE options can be modified using `ore.hiveOptions`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

`ore.hiveOptions`

Examples

```
if(ore.is.connected(type="HIVE"))
{
ore.showHiveOptions()
}
```

---

ore.showImpalaOptions *Show Impala Options*

Description

Shows the current value of all Impala options, namely, field delimiters for the Impala tables and the database name.

Usage

```
ore.showImpalaOptions()
```

Details

This function displays the values of Impala options set in the current environment. The Impala options can be modified using `ore.impalaOptions`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

See Also

`ore.impalaOptions`

Examples

```
if(ore.is.connected(type="IMPALA"))
{
ore.showImpalaOptions()
}
```

---

ore.sort *Oracle R Enterprise Data Sorting*

Description

Performs flexible sorting on `ore.frame` objects.

Usage

```
ore.sort(data, by, nls.sort = NULL, reverse = FALSE, stable = FALSE,
         unique.keys = FALSE, unique.data = FALSE,
         cache = getOption(".ore.persist", TRUE))
```

Arguments

data	An ore.frame object.
by	A comma-separated character string specifying the columns to use in sorting from argument data.
nls.sort	A character string specifying Oracle Database NLS_SORT options.
reverse	A logical vector indicating the use of ascending or descending sorts.
stable	A logical value indicating whether to maintain the relative order within sorted groups.
unique.keys	A logical value indicating whether to maintain a single row for each distinct combination of sorting columns.
unique.data	A logical value indicating whether to maintain a single row for each distinct combination of all the columns in argument data.
cache	For internal use only.

Details

Typical use case for ore.sort is during data pre-processing to obtain top-k rows.

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.rank

Examples

```
IRIS <- ore.push(iris)

# Sort all specified columns in descending order
ore.sort(data = IRIS, by = c("Petal.Length", "Sepal.Length"), reverse = TRUE)

# Sort one of the columns in ascending and another in descending order
ore.sort(data = IRIS, by = c("-Petal.Length", "Sepal.Length"))

# Retain just one row per unique value of Petal.Length
ore.sort(data = IRIS, by = "Petal.Length", unique.key = TRUE)

# Remove duplicate rows and rows with the same value for Petal.Length
ore.sort(data = IRIS, by = "Petal.Length", unique.key = TRUE,
         unique.data = TRUE)
```

ore.summary *Oracle R Enterprsie Data Summary*

Description

Generates descriptive statistics for ore.frame objects within flexible row aggregations.

Usage

```
ore.summary(data, var, stats = c("n", "mean", "min", "max"),
            class = NULL, types = NULL, ways = NULL, weight = NULL,
            order = NULL, maxid = NULL, minid = NULL, mu = 0,
            no.type = FALSE, no.freq = FALSE)
```

Arguments

data	An ore.frame object of data.
var	A vector of character strings specifying the names of numeric columns in argument data to which to apply all of the statistical calculations in argument stats, or a list of character string vectors. If the var argument is a list, then the length of the list must be either 1 or the same as the length of stats. If it's a list of length 1, it's equivalent to a vector of strings. If it's a list of length greater than 1, each element of the var list specifies the columns of data to which to apply the statistical calculation in the corresponding position in stats.
stats	A vector of character strings specifying the statistical calculations for argument var. If the name of the vector element is specified, the name becomes the output column name. The values of this argument can be one or more of the following: "n" or "freq" (Count of non-missing values), "count" or "cnt" (Count of all observations), "nmiss" (Count of missing values), "mean" or "avg" (Average of values), "min" (Minimum of values) "max" (Maximum of values), "css" (Corrected sum of squares), "uss" (Uncorrected sum of squares), "cv" (Coefficient of variation), "sum" (Sum of values), "sumwgt" (Weighted sum of values), "range" (Range of values), "stddev" or "std" (Standard deviation of values), "stderr" or "stdmean" (Standard error for the mean), "variance" or "var" (Variance of values), "kurtosis" or "kurt" (Kurtosis), "skewness" or "skew" (Skewness), "loccount<" or "loc<" (Number of observations whose values are less than the supplied mu),

"loccount>" or "loc>" (Number of observations whose values are greater than the supplied mu),

"loccount!" or "loc!" (Number of observations whose values are not equal to the supplied mu),

"loccount" or "loc" (Number of observations whose values are equal to the supplied mu),

Percentiles Types: "p0", "p1", "p5", "p10", "p25" or "q1", "p50" or "q2" or "median", "p75" or "q3", "p90", "p95", "p99", "p100" (Percentile or quantile),

"qrange" or "iqr" (Interquartile range, Q3-Q1),

"mode" (Most frequently occurring value),

"lclm" (Two-sided left confidence limit with confidence level of the interval equal to 0.95),

"rclm" (Two-sided right confidence limit with confidence level of the interval equal to 0.95),

"clm" (Two-sided confidence interval with confidence level of the interval equal to 0.95),

"t" (Student's t-test statistic),

"probt" or "prt" (Two-tailed p-value for student's t-test)

class — A vector of character strings specifying the names of categorical columns within argument data. If not specified, the aggregation of the entire data is returned.

types — A list of character string vectors specifying the combinations of the column names in class within which the aggregations will be executed in the returning summary.

ways — A vector of integers with each value indicating the number of columns in class that are used to generate types. With one integer number, it generates types of all possible combinations with the specified number of columns in class. The types generated by ways will be combined with the types specified in types with redundency removed automatically.

weight — An optional single character string specifying a numeric column within data to use as analytic weights. By default, the weight for each non-missing observation is 1. The statistics in stats that can take weight are "sum", "sumwgt", "mean", "css", "uss", "cv", "stddev", "variance", and "stderr". The weight argument is ignored when specified with other statistics.

order — A vector of character strings specifying the sorting criteria. The values of this argument can be one or more of the following:

"freq" or "-freq" (Ascending or descending sorts based on count statistics),

"type" or "-type" (Ascending or descending sorts based on type),

"class" or "-class" (Ascending or descending sorts based on the columns in class).

maxid — A named vector of character strings, each element of which specifies two columns in data. The name of an element specifies an over-column and the value of the element specifies an id-column. Each element results in an additional column in the returned ore.frame object. Each additional column contains the value from the id-column that corresponds to the observation that has the maximum value in the over-column.

minid	A named vector of character strings, each element of which specifies two columns in data. The name of an element specifies an over-column and the value of the element specifies an id-column. Each element results in an additional column in the returned ore.frame object. Each additional column contains the value from the id-column that corresponds to the observation that has the minimum value in the over-column.
mu	A single number or a vector of numbers whose elements correspond to each value in var, to supply additional numeric parameters for some statistics. The default value is 0. The statistics that use mu are "loccount<", "loccount>", "loccount", "loccount!", "t", and "probt". The mu argument is ignored when specified with other statistics.
no.type	A logical value indicating whether to drop the TYPE column from the output.
no.freq	A logical value indicating whether to drop the FREQ column from the output.

Details

The function ore.summary generates descriptive statistics for ore.frame objects within user specified aggregation sub-groups.

The argument class specifies the columns to be used to define aggregation sub-groups. The arguments types and ways define the sub-groups. If class is NULL, the function aggregates the entire data without sub-groups. If class is specified, but both types and ways are NULL, the function returns aggregations of all possible sub-groups by the columns in class. The number of sub-groups increases exponentially over the number of class columns. Oracle recommands using types and ways to specify the sub-groups of interest.

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Examples

```
IRIS <- ore.push(iris)

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std", "p10"),
  class="Species")

ore.summary(IRIS, list(c("Sepal.Length", "Petal.Length"),
  "Sepal.Width"), c(AVG="mean", "std"), class="Species")

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"),
  class="Species", weight="Sepal.Width")

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"),
  class=c("Species", "Petal.Width"),
```

```
  types=list("Species", c("Species", "Petal.Width")),
  order=c("type", "-freq", "class"))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "std"),
  class=c("Species", "Petal.Width"),
  ways=1, order=c("type", "-freq", "class"))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), c("mean", "prt"),
  class="Species", mu=c(5.8, 3.7))

ore.summary(IRIS, c("Sepal.Length", "Petal.Length"), "mean",
  class="Species",
  maxid=c(Sepal.Length="Sepal.Width", Petal.Length="Petal.Width"))
```

`ore.sync` *Oracle R Enterprise Object Synchronization Function*

Description

Synchronize `ore.frame` objects the represent database tables, views, and queries within the R `environment` for a schema in the Oracle R Enterprise session.

Usage

```
ore.sync(schema, table = NULL, use.keys = TRUE, query = NULL)
```

Arguments

`schema`	A character string specifying the database schema name. Not supported with use of argument `query`.
`table`	An optional character vector specifying the database table or view names to use for `ore.frame` object creation.
`use.keys`	A logical value specifying whether primary keys, if they exist, should be used for constructing ordered `ore.frame` objects.
`query`	An optional named character vector specifying queries to use for `ore.frame` object creation. The element names are used as the names of the `ore.frame` objects in the R `environment` for the default schema.

Details

Oracle R Enterprise creates proxy objects in R that correspond to the tables or views in the database schema. These proxy objects contain metadata used by Oracle R Enterprise internally to provide transparency layer functionality.

By default, function `ore.sync` uses the schema given at the time of `ore.connect`. The `schema` argument can be used to select a different schema for which the user has the appropriate access privileges.

When both the `table` and `query` arguments are `NULL` in an `ore.sync` function call, all tables and views whose name do not contain a `$` or begin with `SYS_` are selected. Selecting all of the tables in a database schema can be expensive and so the recommended practice is to set either the `table` or `query` argument to limit the the number of tables and views represented in the Oracle R Enterprise schema environment.

If argument use.keys is set to FALSE or if the table has no primary key, the corresponding ore.frame object will be unordered; otherwise it will be ordered. For many operations on ore.frame objects, for example colMeans, the row order is not important. However for operations that require a row ordering, for example diff, an error message is issued specifying that the ordering information is not available. A row ordering can be imposed on an ore.frame object by setting its row names to one or more of its columns.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.attach, ore.connect, ore.exists, ore.get, ore.ls, ore.rm, row.names<-

Examples

```
if (!interactive())
{
  ore.sync()
  ore.sync("RQUSER", use.keys = FALSE)
  ore.exec("create table TABLE1 as select * from dual")
  ore.exec("create table TABLE2 as select * from dual")
  ore.sync(table = c("TABLE1", "TABLE2"))
  ore.sync("RQUSER", table = c("TABLE1", "TABLE2"))
  ore.sync(query = c("QUERY1" = "select 0 X, 1 Y from dual",
                     "QUERY2" = "select 1 X, 0 Y from dual"))
  ore.drop("TABLE1")
  ore.drop("TABLE2")
}
```

---

ore.toXML — *Oracle R Enterprise XML String Generation Function*

Description

Creates a string containing an XML representation for vector, matrix, data.frame, list or ore objects.

Usage

```
ore.toXML(obj)
```

Arguments

obj	A vector, matrix, data.frame, list or ore object.

Details

Generates XML strings intended for consumption by external tools, such as Business Intelligence (BI) web portals.

Value

A character string containing an XML representation of `obj`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore`

Examples

```
ore.toXML(1:10)
irisXML <- ore.toXML(iris)
substring(irisXML, 1, 65)
```

---

ore.univariate *Oracle R Enterprise Univariate Summaries*

Description

Performs distribution analysis of numeric columns in an object of type `ore.frame`. Reports all statistics from the function `ore.summary` plus signed-rank test and extreme values.

Usage

```
ore.univariate(data, var, class = NULL, stats = NULL, special = NULL,
               id = NULL, weight = NULL, extremes = FALSE,
               nextrval = 5, exclnpwgt = FALSE, freq = NULL,
               idout = FALSE, loccount = FALSE, mu0 = NULL,
               def = NULL)
```

Arguments

`data`	An `ore.frame` object.
`var`	A comma-separated character string specifying the names of numeric columns within argument `data`.
`class`	A comma-separated character string specifying the names of categorical columns within argument `data`.

stats	A comma-separated character string specifying the statistical calculations for argument var. Currently the supported arguments are moments or m, measures or s, quantiles or q, location or l, normality or n, loccount or lc, extremes or x.
special	Argument not supported.
id	Argument not supported.
weight	An optional character string specifying a numeric column within argument data to use as analytic weights.
extremes	A logical value indicating whether to report extreme values for columns specified in argument var.
nextrval	An integer value specifying the depth of reporting for extreme values; default is 5.
exclnpwgt	Argument not supported.
freq	Argument not supported.
idout	Argument not supported.
loccount	Argument not supported.
mu0	Argument not supported.
def	Argument not supported.

Value

Returns an ore.frame object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.summary

Examples

```
IRIS <- ore.push(iris)

# Default univariate statistics
x <- ore.univariate(IRIS, var=c("Sepal.Length", "Sepal.Width"))

# Compute location statistics on Sepal.Length
x <- ore.univariate(IRIS, var="Sepal.Length", stats="location")

# Compute quantiles statistics on Sepal.Length and Sepal.Width
x <- ore.univariate(IRIS, var=c("Sepal.Length", "Sepal.Width"),
                    stats="quantiles")
```

ore.vector-aggregate

Oracle R Enterprise Univariate Aggregation by a Set of Grouping Variables

Description

Univariate aggregations based on a set of grouping variables.

Usage

```
  ## S4 method for signature 'ore.vector'
aggregate(x, by, FUN, ..., simplify = TRUE)
```

Arguments

x	An ore.vector object.
by	Either an ore.vector object representing a single grouping variable or an ore.frame or list object containing a set of grouping variables.
FUN	One of all, any, fivenum, IQR, length, max, mean, median, min, prod, quantile, range, sd, sum, summary, var.
...	Optional additional arguments supplied to argument FUN.
simplify	Argument not supported.

Details

Unlike the aggregate method for vectors in the **stats** package, the FUN argument is limited to the predefined list given above.

Value

An ore.frame object with (number of grouping variables plus number of aggregate values) columns, where the first set of columns provide the distinct combinations of by argument values and the last set of columns contains the FUN aggregate values.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

aggregate, ore.vector-ave, ore.number-univar

Examples

```
TG <- ore.push(ToothGrowth)
aggregate(TG$len, TG[2:3], mean)
aggregate(TG$len, TG[2:3], quantile)
```

ore.vector-ave *Oracle R Enterprise Univariate Operations within a Set of Grouping Variables*

Description

Performs univariate operations within a set of grouping variables.

Usage

```
  ## S4 method for signature 'ore.vector'
ave(x, ..., FUN = mean)
```

Arguments

x	An ore.vector object.
...	A set of ore.vector and ore.frame objects representing grouping variables as well as optional arguments to FUN.
FUN	One of length, max, mean, median, min, rank, sd, sum, var.

Details

Unlike the ave method for vectors in the **stats** package, the FUN argument is limited to the predefined list given above.

Value

An ore.vector object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ave, ore.vector-aggregate, ore.number-univar

Examples

```
TG <- ore.push(ToothGrowth)
ave(TG$len, TG[2:3], FUN = mean)
ave(TG$len, TG[2:3], FUN = rank)
```

ore.vector-class *Class* ore.vector

Description

The ore.vector class represents data columns in Oracle R Enterprise.

Accessors

length(x): Returns the number of elements in argument x.

names(x): Returns an ore.character object containing the element names. When the element names are made of multiple components, they will be separated with the value specified in the ore.sep option, which by default is set to "|".

names(x) <-value: Replaces the element names in argument x with the names in argument value. The value argument must be either NULL to remove element names, an ore.vector object for single component names, or an ore.frame object for multiple component names.

Subsetting

In the code snippets below, argument x is an ore.vector object.

x[i]: Returns the ore.logical selected elements of argument x as an ore.vector object.

x[i] <-value: Creates/replaces the ore.logical specified elements of argument x with value.

head(x,n = 6L): If argument n is non-negative, returns the first n elements of argument x. If argument n is negative, returns all but the last abs(n) elements of argument x.

tail(x,n = 6L): If argument n is non-negative, returns the last n elements of argument x. If n is negative, returns all but the first abs(n) elements of argument x.

Splitting and Combining

In the code snippets below, argument x is an ore.vector object.

split(x,f,drop = FALSE): Splits argument x into a list object, according to argument f, dropping elements corresponding to unrepresented levels if drop is TRUE.

c(...): Returns a new ore.vector object by combining the elements of the ore.vector objects in ....

Looping

In the code snippets below, argument x is an ore.frame object.

tapply(X,INDEX,FUN = NULL,...,simplify = TRUE): Apply argument FUN to each partitioning of argument data, an ore.vector object, specified by the list of factor objects argument INDEX.

by)(data,INDICES,FUN,...,simplify = TRUE): Apply argument FUN to each partitioning of argument data, an ore.vector object, specified by the factor (or list of factor objects) argument INDICES.

Utilities

In the code snippets below, argument x is an ore.vector object.

x %in% table: Returns an ore.logical object indicating which values in argument x are present in argument table. To maximize performance, if table is a large R vector, then use ore.push to convert it to an ore.vector before using x %in% table.

is.na(x): Returns an ore.logical object indicating which values in argument x contain missing values.

pmax(...,na.rm = FALSE): Parallel maxima of the ore.vector objects in the ... argument. Removes NA values when na.rm = TRUE.

pmin(...,na.rm = FALSE): Parallel minima of the ore.vector objects in the ... argument. Removes NA values when na.rm = TRUE.

rank(x,na.last = TRUE,ties.method = c("average","first","random","max","min")): Returns the ranks for values of argument x.

sort(x,decreasing = FALSE,na.last = NA,...): Returns a sorted version of argument x.

table(...,exclude,useNA = c("no","ifany","always"),dnn,deparse.level = 1): Tabulates the values of argument x.

unique(x): Returns the unique values of argument x.

Group Generics

ore.vector objects have support for S4 group generic functionality:

Compare "==", ">", "<", "!=", "<=", ">="

Summary "max", "min", "range", "prod", "sum", "any", "all"

See S4groupGeneric for more details.

Coercion

In the code snippets below, argument x is an ore.vector object.

as.character(x): Returns an ore.character object coercion of argument x.

as.data.frame(x,row.names = NULL,optional = FALSE,...,nm = deparse(substitute(x,par = 30L)[1L],stringsAsFactors = default.stringsAsFactors()): Returns a single column ore.frame object containing the values of argument x.

as.factor(x): Returns an ore.factor object coercion of argument x.

as.integer(x): Returns an ore.integer object coercion of argument x.

as.logical(x): Returns an ore.factor object coercion of argument x.

as.matrix(x,...): Returns a single column ore.matrix object containing the values of argument x.

as.numeric(x): Returns an ore.numeric object coercion of argument x.

as.vector(x,mode = "any"): Returns an ore.vector object based on the values contained in argument x. The vector will be coerced to the requested mode, unless argument mode is "any", in which case the most appropriate type is chosen.

Note

See the corresponding R documentation for the functions listed above.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore, ore.character, ore.datetime, ore.factor, ore.frame, ore.logical, ore.matrix, ore.number

Examples

```
showClass("ore.vector")
```

---

ore.year *Oracle R Enterprise Date/Time Functions*

Description

Extracts date and time related information from date/time data types.

Usage

```
ore.year(x, ...)
ore.month(x, ...)
ore.mday(x, ...)
ore.hour(x, ...)
ore.minute(x, ...)
ore.second(x, ...)
```

Arguments

x	For functions ore.year and ore.month, ore.date and ore.datetime objects. For function ore.mday, ore.date, ore.datetime, ore.difftime and objects. For functions ore.hour, ore.minute, and ore.second, ore.datetime and ore.difftime objects.
...	For future expansion.

Details

These functions return the following information:

ore.year Year with century.

ore.month Month (1-12; January = 1).

ore.mday Day of month (1-31).

ore.hour Hour of day (0-23).

ore.minute Minute (0-59).

ore.second Second (0-59).

Value

For functions ore.year, ore.month, ore.mday, ore.hour, and ore.minute, returns an ore.integer object.

For function ore.second, returns an ore.numeric object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.date, ore.datetime, ore.difftime

Examples

```
DATE <- ore.push(seq(as.Date("1999/3/14"), as.Date("2002/2/16"), by="3 months"))
DATE
ore.year(DATE)
ore.month(DATE)
ore.mday(DATE)
```

OREDistributions *Oracle R Enterprise Distribution Functions*

Description

Density, cumulative distribution function, quantile function and random variate generation for many standard probability distributions.

Details

The functions for the density/mass function, cumulative distribution function, and quantile function are named in the form dxxx, pxxx, and qxxx respectively.

Refer to dbeta for information on the beta distribution.

Refer to dbinom for information on the binomial (including Bernoulli) distribution .

Refer to dcauchy for information on the Cauchy distribution.

Refer to dchisq for information on the chi-squared distribution.

Refer to dexp for information on the exponential distribution.

Refer to df for information on the F distribution.

Refer to dgamma for information on the gamma distribution.

Refer to dgeom for information on the geometric distribution. (This is also a special case of the negative binomial.)

Refer to dlnorm for information on the log-normal distribution.

Refer to dlogis for information on the logistic distribution.

Refer to dnbinom for information on the negative binomial distribution.

Refer to dnorm for information on the normal distribution.

Refer to dpois for information on the Poisson distribution.

Refer to dsignrank for information on the Wilcoxon Signed Rank statistic distribution.

Refer to dt for information on Student's t distribution.

Refer to dunif for the uniform distribution.

Refer to dweibull for information on the Weibull distribution.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

ore.number

OREShowDoc *Show Oracle R Enterprise Manuals and Other Documentation*

Description

Utility function that displays the Oracle R Enterprise documentation welcome page in a web browser.

Usage

```
OREShowDoc()
```

Details

An Oracle R Enterprise welcome page will be displayed in a web browser containing references to Oracle R Enterprise Release Notes as well as Oracle R Enterprise User's Guide.

Value

Returns an invisible character string specifying the path to the Oracle R Enterprise welcome page.

See Also

RShowDoc

Examples

```
## Not run:
OREShowDoc()

## End(Not run)
```

pairs.ore.frame *Scatterplot Matrices*

Description

Creates a matrix of scatterplots.

Usage

```
  ## S3 method for class 'ore.frame'
pairs(x, labels, panel = points, ...,
        lower.panel = panel, upper.panel = panel,
        diag.panel = NULL, text.panel, label.pos,
        cex.labels = NULL, font.labels = 1,
        row1attop = TRUE, gap = 1)
```

Arguments

x	An ore.frame object.
labels,panel,...,lower.panel,upper.panel,diag.panel,text.panel,label.pos,cex.lab	See documentation in pairs.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

pairs

partitions *Partitions and Settings of an Oracle Data Mining Model Object*

Description

The partition function returns partitions names from a partitioned model. The settings function returns the Oracle Data Mining parameter settings used to build the model.

Usage

```
partitions(object)

## S3 method for class 'ore.odmPart'
partitions(object)

settings(object)

## S3 method for class 'ore.model'
settings(object)
```

Arguments

object	An partitioned model object.

Value

The function partitions returns an ore.frame listing each partition of the specified model object with the associated partition column values of the model. The partition name is system-determined. The function returns NULL for a non-partitioned model.

The function settings returns a data.frame listing each Oracle Data Mining parameter setting name and value pair used to build the model.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmAssocRules, ore.odmKMeans, ore.odmOC,

Examples

```
irisp <- iris
irisp[, "part"] <-  as.numeric(rownames(iris))%%2
IRISP <- ore.push(irisp)

# build partition model with one partition column
svm.pmod <- ore.odmSVM(Sepal.Length ~ ., data = IRISP, type = "regression",
              odm.settings = list(odms_partition_columns = "Species"))

settings(svm.pmod)
summary(svm.pmod)
partitions(svm.pmod)
head(predict(svm.pmod, IRISP, supplemental.cols = "part"))
```

```
# build partition model with two partition columns
svm.pmod2 <- ore.odmSVM(Sepal.Length ~ ., data = IRISP, type = "regression",
      odm.settings = list(odms_partition_columns = c("Species", "part")))

settings(svm.pmod2)
summary(svm.pmod2)
partitions(svm.pmod2)
head(predict(svm.pmod2, IRISP, supplemental.cols = "part"))
```

`plot.ore.vector` *Generic X-Y Plotting*

Description

Generic plotting function.

Usage

```
  ## S3 method for class 'ore.vector'
plot(x, y = NULL, type = "p",
       xlim = NULL, ylim = NULL, log = "",
       main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
       ann = par("ann"), axes = TRUE, frame.plot = axes,
       panel.first = NULL, panel.last = NULL, asp = NA, ...)
```

Arguments

`x,y`	`ore.vector` objects.
`type,xlim,ylim,log,main,sub,xlab,ylab,ann,axes,frame.plot,panel.first,panel.last`	See description in `plot`.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`plot`

points.ore *Add Points to a Plot*

Description

Draws points.

Usage

```
  ## S3 method for class 'ore'
points(x, y = NULL, type = "p", ...)
```

Arguments

x,y	ore.number objects.
type,...	See description in points.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

points

prop.test-methods *Oracle R Enterprise Test of Equal or Given Proportions*

Description

Tests the null hypothesis that either the proportions in several groups are the same, or that they equal certain given values.

Usage

```
  ## S4 method for signature 'ore.vector'
prop.test(x, n, p = NULL,
          alternative = c("two.sided", "less", "greater"),
          conf.level = 0.95, correct = TRUE)
```

Arguments

x	An ore.vector object.
n	Argument not supported.
p	A vector of probabilities; one for each distinct value in argument x.
alternative	A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
conf.level	A numeric value specifying the level of the confidence interval.
correct	A logical value indicating whether Yates' continuity correction should be applied where possible.

Value

Returns an prop.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Wilson, E.B. (1927) Probable inference, the law of succession, and statistical inference. *J. Am. Stat. Assoc.*, **22**, 209–212.

Newcombe R.G. (1998) Two-Sided Confidence Intervals for the Single Proportion: Comparison of Seven Methods. *Statistics in Medicine* **17**, 857–872.

Newcombe R.G. (1998) Interval Estimation for the Difference Between Independent Proportions: Comparison of Eleven Methods. *Statistics in Medicine* **17**, 873–890.

Oracle R Enterprise

See Also

prop.test

---

ranking *Various Ranking Functions*

Description

Six variations of ranking functions to rank the elements in an ordered ore.vector by its values. An ore.character is coerced to an ore.factor. The values of ore.factor are based upon factor levels. Use desc to reverse the direction.

Usage

```
row_number(x)

ntile(x, n)

min_rank(x)

dense_rank(x)

percent_rank(x)

cume_dist(x)
```

Arguments

`x`	An ordered `ore.number`, `ore.character`, or `ore.factor` object to rank. Missing values are left as is.
`n`	The number of groups to split into.

Details

- `row_number`: Equivalent to `rank(ties.method = "first")`.
- `min_rank`: Equivalent to `rank(ties.method = "min")`.
- `dense_rank`: Like `min_rank`, but with no gaps between ranks.
- `percent_rank`: A number between 0 and 1 computed by rescaling `min_rank` to [0, 1].
- `cume_dist`: A cumulative distribution function. The proportion of all values that are less than or equal to the current rank.
- `ntile`: A rough rank, which breaks the input vector into `n` buckets.

Examples

```
X <- ore.push(c(5, 1, 3, 2, 2, NA))

row_number(X)
row_number(desc(X))

min_rank(X)

dense_rank(X)

percent_rank(X)

cume_dist(X)

ntile(X, 2)
ntile(ore.push(runif(100)), 10)
```

`reorder` *Reorder Levels of an* `ore.factor` *Object*

Description

Reorders the levels of an `ore.factor` based on the values of a second `ore.vector`.

Usage

```
  ## S4 method for signature 'ore.factor'
reorder(x, X, FUN = mean, ...,
          order = is.ordered(x))
```

Arguments

`x`	An `ore.factor` object.
`X`	An `ore.vector` of the same length as argument `x`.
`FUN`	A `function` whose first argument represents an `ore.vector` and returns a scalar, to be applied to each subset of argument `X` determined by the levels of argument `x`.
`...`	Optional additional arguments supplied to argument `FUN`.
`order`	Argument not implemented.

Value

Returns an `ore.factor` object with reordered factor levels.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

`ore.factor`

Examples

rules *Predicate Rule Data for Clusters or Association Rules Models*

Description

The generic rules function is used to return cluster predicate rule data from a clustering model or to return association rules from an association model.

Usage

```
rules(object,...)

## S3 method for class 'ore.odmAssocRules'
rules(object, ...)
## S3 method for class 'ore.odmKMeans'
rules(object,...)
## S3 method for class 'ore.odmOC'
rules(object,...)
## S3 method for class 'ore.odmEM'
rules(object,...)
```

Arguments

object	An object for which cluster rules or association rules are desired.
...	Additional arguments affecting the result.

Details

For clustering models (ore.odmKMeans, ore.odmOC), and ore.odmEM, the function rules provides the predicate rule data that defines each cluster as determined by the algorithm.

For an association model (ore.odmAssocRules), the function rules provides the property of each association rule.

Value

For clustering model objects, the function rules returns a list with elements for each cluster. Each element contains a data.frame with the following columns (note that common data repeats across rows):

rhs.cluster.id	The numeric identifier associated with the cluster.
rhs.support	The right-hand side number of rows (support) for this cluster.
rhs.conf	The right-hand side rule confidence associated with this cluster.
lhs.conf	The left-hand side rule confidence associated with this cluster.
lhs.variable	The left-hand side variable name.
lhs.var.support	The left-hand side number of rows (support) for the variable and the combined predicate.
lhs.var.conf	The left-hand side confidence for variable and combined predicate.

predicate — For numerical variables, the lower or upper bound predicate, for categorical variables, the specific category.

For association model objects, the function rules returns an object of class ore.rules that describes the property of each rule. An ore.rules object has the following columns:

RULE_ID — The numerical identifier associated with each rule.

NUMBER_OF_ITEMS — The number of items in the rule.

LHS — The itemset of the antecedent.

RHS — The itemset of the consequent.

SUPPORT — The support of the rule.

CONFIDENCE — The confidence of the rule.

LIFT — The lift of the rule.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmAssocRules, ore.odmKMeans, ore.odmOC, ore.odmEM, ore.rules-class

Examples

```
# clustering
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
         matrix(rnorm(100, mean = 2, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")

X <- ore.push (data.frame(x))
km.mod <- NULL
km.mod <- ore.odmKMeans(~., X, num.centers=2)
rules.km <- rules(km.mod)

oc.mod <- NULL
oc.mod <- ore.odmOC(~., X, num.centers=2)
rules.oc <- rules(oc.mod)

# association rules
id <- c(1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3)
item <- c("b", "d", "e", "a", "b", "c", "e", "b", "c", "d", "e")
data.ore <- ore.push(data.frame(ID = id, ITEM = item))

ar.mod <- ore.odmAssocRules(~., data.ore, case.id.column = "ID",
item.id.column = "ITEM", min.support = 0.6, min.confidence = 0.6,
```

```
  max.rule.length = 3)

  rules.ar <- rules(ar.mod)
```

`sample` *Sample Rows from an* `ore.frame` *Object*

Description

Samples an `ore.frame` object by a fixed number of rows or a fraction, with optional arguments of a sample weight and whether or not to use replacement.

Usage

```
sample_n(tbl, size, replace = FALSE, weight = NULL)

sample_frac(tbl, size = 1, replace = FALSE, weight = NULL)
```

Arguments

`tbl`	An `ore.frame` object to sample.
`size`	For `sample_n`, the number of rows to select. For `sample_frac`, the fraction of rows to select. If `tbl` is grouped, then `size` applies to each group.
`replace`	Specify `TRUE` to sample with replacement or `FALSE` to sample without replacement. The default is FALSE.
`weight`	Sampling weights. This expression is evaluated in the context of the input `ore.frame` or an `ore.number` object. It must return a vector of non-negative numbers with the same length as the input. Weights are automatically normalized to sum to 1.

Examples

```
MTCARS <- ore.push(mtcars)
by_cyl <- group_by(MTCARS, cyl)

# Sample fixed number per group
sample_n(MTCARS, 10)
nrow(sample_n(MTCARS, 50, replace = TRUE))
sample_n(MTCARS, 10, weight = mpg)
sample_n(MTCARS, 10, weight = MTCARS[["mpg"]])

arrange(sample_n(by_cyl, 3), cyl, mpg)
arrange(summarise(sample_n(by_cyl, 10, replace = TRUE), n = n()), cyl)
arrange(summarise(sample_n(by_cyl, 3, weight = mpg/mean(mpg)), n = n()), cyl)
arrange(summarise(sample_n(by_cyl, 3, weight = by_cyl[["mpg"]]/mean(by_cyl[["mpg"]])), n

# Sample fixed fraction per group
nrow(sample_frac(MTCARS, 0.1))
nrow(sample_frac(MTCARS, 1.5, replace = TRUE))
nrow(sample_frac(MTCARS, 0.1, weight = 1/mpg))

arrange(summarise(sample_frac(by_cyl, 0.2), n = n()), cyl)
arrange(summarise(sample_frac(by_cyl, 1, replace = TRUE), n = n()), cyl)
```

select	*Select, Rename, Arrange, Filter, Mutate, or Transmute an* ore.frame *Object*

Description

select Selects only the specified columns. rename Renames the specified columns and keeps all columns. arrange Orders rows by the specified columns. filter Filters rows by matching the specified condition. mutate Adds new columns. transmute Adds new columns and drops the existing columns.

Usage

```
select(.data, ...)
select_(.data, ..., .dots)

rename(.data, ...)
rename_(.data, ..., .dots)

arrange(.data, ...)
arrange_(.data, ..., .dots)

filter(.data, ...)
filter_(.data, ..., .dots)

mutate(.data, ...)
mutate_(.data, ..., .dots)

transmute(.data, ...)
transmute_(.data, ..., .dots)
```

Arguments

.data	An ore.frame object.
...	Comma separated list of unquoted expressions. See select, rename, arrange, filter, mutate, transmute for details.
.dots	Used by select_, rename_, arrange_, filter_, mutate, and transmute_ for standard evaluation. See select_, rename_, arrange_, filter_, mutate_, transmute_ for details.

Value

An ore.frame object.

Special functions

select does not support the special functions in select.

See Also

slice, slice_, summarise, summarise_

Examples

```
IRIS <- ore.push(iris) # so it prints a little nicer

# select specified columns
names(select(IRIS, Petal.Length))
names(select(IRIS, petal_length = Petal.Length))

# drop specified column
names(select(IRIS, -Petal.Length))

names(select_(IRIS, ~Petal.Length))
names(select_(IRIS, petal_length = quote(Petal.Length)))
names(select_(IRIS, .dots = list("-Petal.Length")))

# rename() keeps all variables
names(rename(IRIS, petal_length = Petal.Length))

# Programming with select
head(select_(IRIS, ~Petal.Length))
head(select_(IRIS, "Petal.Length"))
head(select_(IRIS, quote(-Petal.Length), quote(-Petal.Width)))
head(select_(IRIS, .dots = list(quote(-Petal.Length), quote(-Petal.Width))))

# arrange ore.frame
MTCARS <- ore.push(mtcars)
arrange(MTCARS, cyl, disp)
arrange(MTCARS, desc(disp))

# filter ore.frame
head(filter(MTCARS, cyl == 8))
head(filter(MTCARS, cyl < 6))

# Multiple criteria
head(filter(MTCARS, cyl < 6 & vs == 1))
head(filter(MTCARS, cyl < 6 | vs == 1))

# Multiple arguments are equivalent to and
head(filter(MTCARS, cyl < 6, vs == 1))

head(mutate(MTCARS, displ_l = disp / 61.0237))
head(transmute(MTCARS, displ_l = disp / 61.0237))

head(mutate(MTCARS, cyl = NULL))
head(mutate(MTCARS, cyl = NULL, hp = NULL, displ_l = disp / 61.0237))
```

`slice` *Select Rows by Positions*

Description

`slice` works with ordered `ore.frame` object. It ignores the grouping of the input `ore.frame`.

Usage

```
slice(.data, ...)

slice_(.data, ..., .dots)
```

Arguments

.data	An ordered ore.frame object.
...	An integer vector specifying row positions.
.dots	Used to work around non-standard evaluation. See slice_ for details.

See Also

Other single.table.verbs: arrange, arrange_; filter, filter_; mutate, mutate_, transmute, transmute_; rename, rename_, select, select_; summarise, summarise_

Examples

```
MTCARS <- ore.push(mtcars)
rownames(MTCARS)

slice(MTCARS, 1L)
slice(MTCARS, n())
slice(MTCARS, 25:n())

MTCARS <- arrange(MTCARS, hp)
slice(MTCARS, 1L)
slice(MTCARS, n())
slice(MTCARS, 25:n())

# grouping is ignored by slice
# use filter and row_number to obtain slices per group
by_cyl <- group_by(MTCARS, cyl)
filter(by_cyl, row_number(hp) < 3L)    # slice(by_cyl, 1:2)
```

---

summarise — *Summarise Columns by Aggregate Functions*

Description

Aggregates the specified column values.

When an ore.frame object is grouped, the aggregate function is applied group-wise. The supported aggregate functions are min, mean, max, median, length, IQR, prod, sum, range, quantile, fivenum, summary, sd, var, all, and any. The resulting ore.frame drops one grouping of the input ore.frame.

Usage

```
summarise(.data, ...)

summarise_(.data, ..., .dots)
```

Arguments

.data	An ore.frame object.
...	Name-value pairs of aggregate functions such as min(), mean(), max(), and so on.
.dots	Used to work around non-standard evaluation. See summarise_ for details.

Value

An ore.frame object.

See Also

arrange, arrange_; filter, filter_; mutate, mutate_, transmute, transmute_; rename, rename_, select, select_; slice, slice_

Examples

```
MTCARS <- ore.push(mtcars)
summarise(MTCARS, mean(disp))
arrange(summarise(group_by(MTCARS, cyl), mean(disp)), cyl)
arrange(summarise(group_by(MTCARS, cyl), m = mean(disp), r = range(disp)), cyl)

arrange(summarise_(group_by(MTCARS, cyl), m = "mean(disp)", r = quote(range(disp))), cyl)

library(magrittr)
by_cyl <- MTCARS %>% group_by(cyl)
by_cyl %>% summarise(a = n(), b = n() + 1) %>% arrange(cyl)
```

---

summary.ore.odmAssocRules

Summarize In-Database Association Models

Description

Methods for class ore.odmAssocRules or summary.ore.odmAssocRules objects.

Usage

```
## S3 method for class 'ore.odmAssocRules'
summary(object,...)

## S3 method for class 'summary.ore.odmAssocRules'
print(x,...)
```

Arguments

object	An object of type ore.odmAssocRules.
x	An object of type summary.ore.odmAssocRules.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmAssocRules summarizes the results produced by the ore.odmAssocRules function.

Value

The function summary.ore.odmAssocRules returns an object of class summary.ore.odmAssocRules, which is a list with the following components:

call	The function call used to build the model with the given arguments
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
rules	An object of type ore.rules describing the association rules from the model built.
itemsets	An object of type ore.itemsets describing the item sets from the model built.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmAssocRules, ore.rules, ore.itemsets

Examples

```
## For examples see \code{\link{ore.odmAssocRules}}.
```

---

summary.ore.odmDT *Summarize In-Database Decision Tree Models*

Description

Methods for class ore.odmDT or summary.ore.odmDT objects.

Usage

```
## S3 method for class 'ore.odmDT'
summary(object,...)

## S3 method for class 'summary.ore.odmDT'
print(x,...)
```

Arguments

object	An object of type ore.odmDT.
x	An object of type summary.ore.odmDT.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmDT summarizes the results produced by the ore.odmDT function.

Value

The function summary.ore.odmDT returns an object of class summary.ore.odmDT, which is a list with the following components:

call	The function call used to build the model with the given arguments.
n	The number of rows in the training data.
costs	A data.frame containing the cost matrix supplied at model build.
nodes	A data.frame with tree node details, including: parent node id, node id, number of rows assigned to that node, predicted value, split predictate, surrogate variables (if applicable), and full split predicates from current node to root node.
settings	A data.frame of settings used to build the model.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmDT

Examples

```
## For examples see \code{\link{ore.odmDT}}.
```

summary.ore.odmEM *Summarize In-Database Expectation Maximization Models*

Description

Methods for class ore.odmEM or summary.ore.odmEM objects.

Usage

```
## S3 method for class 'ore.odmEM'
summary(object,...)

## S3 method for class 'summary.ore.odmEM'
print(x,...)
```

Arguments

object	An object of type ore.odmEM.
x	An object of type summary.ore.odmEM.
...	Additional arguments affecting the summary produced.

Details

The function summary.ore.odmEM summarizes the results produced by the ore.odmEM function.

Value

The function summary.ore.odmEM returns an object of class summary.ore.odmEM, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
centers2	A data.frame that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmEM

Examples

```
## For examples see \code{\link{ore.odmEM}}.
```

---

summary.ore.odmESA *Summarize In-Database Explicit Semantic Analysis Models*

Description

Methods for class ore.odmESA or summary.ore.odmESA objects.

Usage

```
## S3 method for class 'ore.odmESA'
summary(object,...)

## S3 method for class 'summary.ore.odmESA'
print(x,...)
```

Arguments

object	An object of type ore.odmESA.
x	An object of type summary.ore.odmESA.
...	Additional arguments affecting the summary produced.

Details

The function summary.ore.odmESA summarizes the results produced by ore.odmESA.

Value

The function summary.ore.odmESA returns an object of class summary.ore.odmESA, which is a list with the following components:

name	The name of the model object.
call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
features	A data.frame describing the features extracted.
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmESA

Examples

```
## For examples see \code{\link{ore.odmESA}}.
```

---

summary.ore.odmESM *Summarize In-Database ESM Models*

Description

Methods for class ore.odmESM or summary.ore.odmESM objects.

Usage

```
## S3 method for class 'ore.odmESM'
summary(object,...)

## S3 method for class 'summary.ore.odmESM'
print(x,...)
```

Arguments

object	An object of type ore.odmESM.
x	An object of type summary.ore.odmESM.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmESM summarize the results produced by ore.odmESM.

Value

The function summary.ore.odmESM returns an object of class summary.ore.odmESM, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
importance	The importance for different attributes
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmESM

Examples

```
## For examples see \code{\link{ore.odmESM}}.
```

---

summary.ore.odmGLM *Summarize In-Database Generalized Linear Models*

Description

Methods for class ore.odmGLM or summary.ore.odmGLM objects.

Usage

```
## S3 method for class 'ore.odmGLM'
summary(object,...)

## S3 method for class 'summary.ore.odmGLM'
print(x, digits = max(3, getOption("digits") - 3),
        signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	An object of class ore.odmGLM.
x	An object of class summary.ore.odmGLM as produced by the summary method.
digits	The number of significant digits to use when printing.
signif.stars	A logical value indicating whether to print 'significance stars' for each coefficient.
...	Additional arguments.

Details

The function print.summary.ore.odmGLM follows the formatting conventions of print.summary.glm.

Value

The function `summary.ore.odmGLM` returns an object of class `summary.ore.odmGLM`, which is a list with the following components:

`call`	The matched call.
`terms`	The `terms` object used.
`type`	The type of model fit.
`family`	A list containing the element family.
`deviance`	Minus twice the maximized log-likelihood, up to a constant.
`aic`	The same version of Akaike's *An Information Criterion* as used by `glm`.
`df.residual`	The residual degrees of freedom.
`null.deviance`	The deviance for the null (intercept only) model.
`df.null`	The residual degrees of freedom for the null model.
`na.action`	The number of rows with missing values that were removed.
`deviance.resid`	The deviance residuals.
`coefficients`	The matrix of coefficients, standard errors, z-values and p-values.
`aliased`	The named logical vector showing if the original coefficients are aliased (all elements are `FALSE`).
`dispersion`	The estimated dispersion.
`df`	A 3-vector of the rank of the model and the number of residual degrees of freedom, plus number of non-aliased coefficients.
`nonreference`	For logistic regression, the response values that represents success.
`ridge`	The `ridge` argument.
`auto.data.prep`	The `auto.data.prep` argument.

The return value for linear regression models also include elements:

`residuals`	The deviance residuals.
`sigma`	The square root of the estimated variance of the random error.
`r.squared`	The R-squared value.
`adj.r.squared`	The adjusted R-squared value.
`fstatistic`	A three element vector containing the value of the F-statistic with its numerator and denominator degrees of freedom.

Author(s)

Oracle `<oracle-r-enterprise@oracle.com>`

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmGLM

Examples

```
## For examples see \code{\link{ore.odmGLM}}.
```

---

summary.ore.odmKMeans

Summarize In-Database Hierarchical K-Means Models

Description

Methods for class ore.odmKMeans or summary.ore.odmKMeans objects.

Usage

```
## S3 method for class 'ore.odmKMeans'
summary(object,...)

## S3 method for class 'summary.ore.odmKMeans'
print(x,...)
```

Arguments

object	An object of type ore.odmKMeans.
x	An object of type summary.ore.odmKMeans.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmKMeans summarizes the results produced by the ore.odmKMeans function.

Value

The function summary.ore.odmKMeans returns an object of class summary.ore.odmKMeans, which is a list with the following components:

call	The function call used to build the model with the given arguments
settings	A data.frame of settings used to build the model.
centers2	A data.frame that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmKMeans

Examples

```
## For examples see \code{\link{ore.odmKMeans}}.
```

---

summary.ore.odmNB *Summarize In-Database Naive Bayes Models*

Description

Methods for class ore.odmNB or summary.ore.odmNB objects.

Usage

```
## S3 method for class 'ore.odmNB'
summary(object,...)

## S3 method for class 'summary.ore.odmNB'
print(x,...)
```

Arguments

object	An object of type ore.odmNB.
x	An object of type summary.ore.odmNB.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmNB summarize the results produced by ore.odmNB.

Value

The function summary.ore.odmNB returns an object of class summary.ore.odmNB, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
apriori	The class distribution for the dependent variable.
tables	A list of table objects, one for each predictor variable with conditional probabilities.
levels	A vector of unqiue target class values.
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmNB

Examples

```
## For examples see \code{\link{ore.odmNB}}.
```

---

summary.ore.odmNMF *Summarize In-Database Non-Negative Matrix Factorization Models*

Description

Methods for class ore.odmNMF or summary.ore.odmNMF objects.

Usage

```
## S3 method for class 'ore.odmNMF'
summary(object,...)

## S3 method for class 'summary.ore.odmNMF'
print(x,...)
```

Arguments

object	An object of type ore.odmNMF.
x	An object of type summary.ore.odmNMF.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmNMF summarizes the results produced by ore.odmNMF.

Value

The function summary.ore.odmNMF returns an object of class summary.ore.odmNMF, which is a list with the following components:

name	The name of the model object.
call	The function call used to build the model with the given arguments
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
features	A data.frame describing the features extracted.
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmNMF

Examples

```
## For examples see \code{\link{ore.odmNMF}}.
```

---

summary.ore.odmNN *Summarize In-Database Neural Network Models*

Description

Methods for class ore.odmNN or summary.ore.odmNN objects.

Usage

```
## S3 method for class 'ore.odmNN'
summary(object,...)

## S3 method for class 'summary.ore.odmNN'
print(x,...)
```

Arguments

object	An object of type ore.odmNN.
x	An object of type summary.ore.odmNN.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmNN summarize the results produced by ore.odmNN.

Value

The function summary.ore.odmNN returns an object of class summary.ore.odmNN, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
importance	The importance for different attributes
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmNN

Examples

```
## For examples see \code{\link{ore.odmNN}}.
```

---

summary.ore.odmOC *Summarize In-Database Orthogonal Partitioning Cluster Models*

Description

Methods for class ore.odmOC or summary.ore.odmOC objects.

Usage

```
## S3 method for class 'ore.odmOC'
summary(object,...)

## S3 method for class 'summary.ore.odmOC'
print(x,...)
```

Arguments

object	An object of type ore.odmOC.
x	An object of type summary.ore.odmOC.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmOC summarizes the results produced by ore.odmOC.

Value

The function summary.ore.odmOC returns an object of class summary.ore.odmOC, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
clusters	A data.frame with each row describing a cluster.
centers2	A data.frame that has the cluster ID values as rownames, and columns corresponding to variable means (if numerical) or mode (if categorical).
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmOC

Examples

```
## For examples see \code{\link{ore.odmOC}}.
```

summary.ore.odmRF *Summarize In-Database Neural Network Models*

Description

Methods for class ore.odmRF or summary.ore.odmRF objects.

Usage

```
## S3 method for class 'ore.odmRF'
summary(object,...)

## S3 method for class 'summary.ore.odmRF'
print(x,...)
```

Arguments

object	An object of type ore.odmRF.
x	An object of type summary.ore.odmRF.
...	Additional arguments affecting the predictions produced.

Details

The function summary.ore.odmRF summarize the results produced by ore.odmRF.

Value

The function summary.ore.odmRF returns an object of class summary.ore.odmRF, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
attributes	A data.frame of attributes used to build the model.
importance	The importance for different attributes
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmRF

Examples

```
## For examples see \code{\link{ore.odmRF}}.
```

`summary.ore.odmSVD` *Summarize In-Database Singular Value Decomposition Models*

Description

Methods for class `ore.odmSVD` or `summary.ore.odmSVD` objects.

Usage

```
## S3 method for class 'ore.odmSVD'
summary(object,...)

## S3 method for class 'summary.ore.odmSVD'
print(x,...)
```

Arguments

`object`	An object of type `ore.odmSVD`.
`x`	An object of type `summary.ore.odmSVD`.
`...`	Additional arguments affecting the summary produced.

Details

The function `summary.ore.odmSVD` summarizes the results produced by `ore.odmSVD`.

Value

The function `summary.ore.odmSVD` returns an object of class `summary.ore.odmSVD`, which is a list with the following components:

`name`	The name of the model object.
`call`	The function call used to build the model with the given arguments.
`settings`	A `data.frame` of settings used to build the model.
`attributes`	A `data.frame` of attributes used to build the model.
`d`	An `ore.frame` with the sigular values of the input data. Column `FEATURE_ID` represents the sigular value ID.
`v`	An `ore.frame` whose columns contain the right singular vectors. The column names match with `FEATURE_ID` values.
`u`	An `ore.frame` whose columns contain the left singular vectors.
`features`	An `ore.frame` same as `v` but in unpivot format.
`formula`	The `formula` used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVD

Examples

```
## For examples see \code{\link{ore.odmSVD}}.
```

---

summary.ore.odmSVM *Summarize In-Database Support Vector Machine Models*

Description

Methods for class ore.odmSVM or summary.ore.odmSVM objects.

Usage

```
## S3 method for class 'ore.odmSVM'
summary(object,...)

## S3 method for class 'summary.ore.odmSVM'
print(x,...)
```

Arguments

object	An object of type ore.odmSVM.
x	An object of type summary.ore.odmSVM.
...	Additional arguments affecting the predictions produced.

Details

The function print.summary.ore.odmSVM tries to be smart about formatting the coefficients, standard errors, and so on.

Value

The function summary.ore.odmSVM returns an object of class summary.ore.odmSVM, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
fit.values	An link[OREbase]{ore.frame} of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL', 'PREDICTED', 'PROBABILITY'. For anomaly detection, the columns are 'PREDICTED' and 'PROBABILITY'.

residuals	For regression models, an link[OREbase]{ore.numeric} vector containing the residual values (PREDICTED - ACTUAL).
coefficients	The coefficients of the SVM model, one for each predictor variable. If auto.data.prep was set to TRUE during model build, coefficients will be in the transformed space (after automatic outlier-aware normalization is applied).
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmSVM

Examples

```
## For examples see \code{\link{ore.odmSVM}}.
```

---

summary.ore.odmXGB *Summarize In-Database XGBoost Models*

Description

Methods for class ore.odmXGB or summary.ore.odmXGB objects.

Usage

```
## S3 method for class 'ore.odmXGB'
summary(object,...)

## S3 method for class 'summary.ore.odmXGB'
print(x,...)
```

Arguments

object	An object of type ore.odmXGB.
x	An object of type summary.ore.odmXGB.
...	Additional arguments affecting the predictions produced.

Details

The function print.summary.ore.odmXGB tries to be smart about formatting the settings, importance, and so on.

Value

The function summary.ore.odmXGB returns an object of class summary.ore.odmXGB, which is a list with the following components:

call	The function call used to build the model with the given arguments.
settings	A data.frame of settings used to build the model.
fit.values	An link[OREbase]{ore.frame} of the actual column and predicted column. For regression, the columns are 'ACTUAL' and 'PREDICTED'. For classification, the columns are 'ACTUAL', 'PREDICTED', 'PROBABILITY'.
residuals	For regression models, an link[OREbase]{ore.numeric} vector containing the residual values (PREDICTED - ACTUAL).
importance	The attribute importance of the XGB model, one for each predictor variable.
formula	The formula used for the symbolic description of the model fitted.

Author(s)

Oracle

Maintainer: Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

Oracle Data Mining Concepts

Oracle Data Mining User's Guide

See Also

ore.odmXGB

Examples

```
## For examples see \code{\link{ore.odmXGB}}.
```

---

sunflowerplot.ore *Produce a Sunflower Scatter Plot*

Description

Creates sunflower plots.

Usage

```
  ## S3 method for class 'ore'
sunflowerplot(x, y = NULL, number, log = "", digits = 6,
              xlab = NULL, ylab = NULL, xlim = NULL, ylim = NULL,
              add = FALSE, rotate = FALSE, pch = 16, cex = 0.8,
              cex.fact = 1.5, col = par("col"), bg = NA, size = 1/8,
              seg.col = 2, seg.lwd = 1.5, ...)
```

Arguments

x, y	ore.number objects.
number, log, digits, xlab, ylab, xlim, ylim, add, rotate, pch, cex, cex.fact, col, bg, size, se	See description in sunflowerplot.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

sunflowerplot

---

svd-methods *Oracle R Enterprise Singular Value Decomposition*

Description

Singular value decomposition of ore.frame and ore.tblmatrix data.

Usage

```
  ## S4 method for signature 'ore.frame'
svd(x, nu, nv, LINPACK = FALSE)
  ## S4 method for signature 'ore.tblmatrix'
svd(x, nu, nv, LINPACK = FALSE)
```

Arguments

x	An ore.frame or ore.tblmatrix object of a numeric matrix.
nu	This argument is not supported, as the function does not return the left singular vector matrix u. See details.
nv	The number of right singular vectors to be computed. The value should be between 0 and ncol(x). If missing, the default value is either the number of rows or the number of columns, whichever is smaller.
LINPACK	See the description of the LINPACK argument in svd.

Details

This function uses a parallel block update algorithm for singular value decomposition of tall and skinny numeric matrix. The degree of parallelism is regulated by the global option ore.parallel. The function returns the singular value vector d and the right singular vector matrix v. The function does not return the left singular vector matrix u, and thus the argument nu is defunct.

Value

Returns a list object containing the d vector and v matrix components of a singular value decomposition of argument x.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

svd

Examples

```
USARRESTS <- ore.push(USArrests)

svd(USARRESTS)
```

---

t.test-methods *Oracle R Enterprise Student's t-Test*

Description

Performs one and two sample t-tests on vectors of data.

Usage

```
  # one or two sample test
  ## S4 method for signature 'vector'
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

  # two sample test
  ## S4 method for signature 'ore.factor'
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)
```

Arguments

x	An ore.number object or an ore.factor object with exactly two levels.
y	An optional ore.number object.
alternative	A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".

mu	A number specifying the true value of the mean (or difference in means if you are performing a two sample test).
paired	A logical value indicating whether to compute a paired t-test.
var.equal	A logical value indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.
conf.level	A numeric value specifying the level of the confidence interval.
...	Additional arguments to be passed to or from methods.

Value

Returns an t.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

t.test

---

tally *Counts or Tallies Rows by Group*

Description

tally is a convenient wrapper for summarise that will either call n or sum(n) depending on whether you're tallying for the first time, or re-tallying. count is similar, but it does the group_by for you. count_ uses vars to specify the groupng column names.

Usage

```
tally(x, wt, sort = FALSE)

count(x, ..., wt = NULL, sort = FALSE)

count_(x, vars, wt = NULL, sort = FALSE)
```

Arguments

x	A grouped ore.frame object to tally or count.
wt	If not specified, tally the number of rows. If specified, perform a weighted tally by summing over the weight values.
sort	Whether to sort output in descending order of n.
..., vars	Columns to group by.

Examples

```
MTCARS <- ore.push(mtcars)
arrange(tally(group_by(MTCARS, cyl)), cyl)
tally(group_by(MTCARS, cyl), sort = TRUE)

# Multiple tallys progressively roll up the groups
cyl_by_gear <- tally(group_by(MTCARS, cyl, gear), sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

cyl_by_gear <- tally(group_by(MTCARS, cyl, gear), wt = hp,  sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

cyl_by_gear <- count(MTCARS, cyl, gear, wt = MTCARS[["hp"]] + mpg, sort = TRUE)
tally(cyl_by_gear, sort = TRUE)
tally(tally(cyl_by_gear))

library(magrittr)
MTCARS %>% group_by(cyl) %>% tally(sort = TRUE)

# count is more succinct and also does the grouping
MTCARS %>% count(cyl) %>% arrange(cyl)
MTCARS %>% count(cyl, wt = hp) %>% arrange(cyl)
MTCARS %>% count_("cyl", wt = hp, sort = TRUE)
```

---

text.ore *Add Text to a Plot*

Description

Draws text.

Usage

```
  ## S3 method for class 'ore'
text(x, y = NULL, labels = seq_along(x), adj = NULL,
       pos = NULL, offset = 0.5, vfont = NULL, cex = 1, col = NULL,
       font = NULL, ...)
```

Arguments

x,y	ore.number objects.
labels,adj,pos,offset,vfont,cex,col,font,...	See description in text.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

text

---

top_n *Select Top n Rows*

Description

This is a convenient wrapper that uses filter and min_rank to select the top n entries in each group, ordered by wt.

Usage

```
top_n(x, n, wt)
```

Arguments

x	The ore.frame object to filter.
n	The number of rows to return. If x is grouped, this is the number of rows per group. May include more than n if there are ties.
wt	An optional argument to specify the column to use for ordering. If not specified, defaults to the last column in x.

Examples

```
MTCARS <- ore.push(mtcars)

# Find top 3 carbs with most hps
carbs <- group_by(MTCARS, carb)
hps <- tally(carbs, hp)
top_n(hps, 3, n)

library(magrittr)
MTCARS %>% group_by(carb) %>% tally(hp) %>% top_n(3)
MTCARS %>% group_by(carb) %>% top_n(1, hp)
```

---

var.test-methods *Oracle R Enterprise F Test to Compare Two Variances*

Description

Performs an F test to compare the variances of two samples from normal populations.

Usage

```
  # one or two sample test
  ## S4 method for signature 'vector'
var.test(x, y, ratio = 1,
         alternative = c("two.sided", "less", "greater"),
         conf.level = 0.95, ...)

  # two sample test
  ## S4 method for signature 'ore.factor'
var.test(x, y, ratio = 1,
         alternative = c("two.sided", "less", "greater"),
         conf.level = 0.95, ...)
```

Arguments

x	An ore.number object or an ore.factor object with exactly two levels.
y	An optional ore.number object.
ratio	A numeric value specifying the hypothesized ratio of the population variances of arguments x and y.
alternative	A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
conf.level	A numeric value specifying the level of the confidence interval.
...	Additional arguments.

Value

Returns an var.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

Oracle R Enterprise

See Also

var.test

---

wilcox.test-methods

Oracle R Enterprise Wilcoxon Rank Sum and Signed Rank Tests

Description

Performs one- and two-sample Wilcoxon tests on vectors of data; the latter is also known as the 'Mann-Whitney' test.

Usage

```
  ## S4 method for signature 'vector'
wilcox.test(x, y = NULL,
            alternative = c("two.sided", "less", "greater"),
            mu = 0, paired = FALSE, exact = NULL, correct = FALSE,
            conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	An ore.number object.
y	An optional ore.number object.
alternative	A character string specifying the alternative hypothesis and must be one of "two.sided" (default), "greater" or "less".
mu	A number specifying an optional parameter used to form the null hypothesis.
paired	A logical value indicating whether to compute a paired test.
exact	Argument not supported.
correct	Argument not supported.
conf.int	Argument not supported.
conf.level	Argument not supported.
...	Additional arguments.

Value

Returns an wilcox.test htest object.

Author(s)

Oracle <oracle-r-enterprise@oracle.com>

References

David F. Bauer (1972), Constructing confidence sets using rank statistics. *Journal of the American Statistical Association* **67**, 687–690.

Myles Hollander and Douglas A. Wolfe (1973), *Nonparametric Statistical Methods.* New York: John Wiley & Sons. Pages 27–33 (one-sample), 68–75 (two-sample).
Or second edition (1999).

Oracle R Enterprise

See Also

wilcox.test

Index